1992

Implementation of a personal computer based parameter estimation program.

Graham, Robert Gordon

Monterey, California. Naval Postgraduate School

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (*If applicable*) AA	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (*City, State, and ZIP Code*) Monterey, CA 93943-5000	7b ADDRESS (*City, State, and ZIP Code*) Monterey, CA 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION 8b OFFICE SYMBOL (*If applicable*)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (*City, State, and ZIP Code*)	10 SOURCE OF FUNDING NUMBERS Program Element No / Project No / Task No / Work Unit Accession Number

11 TITLE (*Include Security Classification*)
Implementation of a Personal Computer Based Parameter Estimation Program

12 PERSONAL AUTHOR(S) Graham, Robert Gordon

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (*year, month, day*)	15 PAGE COUNT
Engineer's Thesis	From To	1992 March	147

16 SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 COSATI CODES			18 SUBJECT TERMS (*continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUBGROUP	Parameter estimation

19 ABSTRACT (*continue on reverse if necessary and identify by block number*)

Aircraft parameter estimation is the process of extracting numerical values for aerodynamic stability and control derivatives from flight-test time history data. This process can be used as a verification or validation tool for results obtained from wind-tunnel testing or through computational analysis, and can obtain or improve estimations of dynamic derivatives. This study implements the MATLAB Personal Computer (PC) based maximum likelihood estimation routine for aircraft longitudinal and lateral-directional derivatives. The parameter estimation was first accomplished on generated simulated data, with and without noise. The noise consisted of measurement and state noise which used the Dryden Gust Model. Secondly, two actual longitudinal flight-test maneuvers are analyzed for the F-14A and the T-37 aircraft. Additionally, the simulated portion of this study can be an excellent instructional aid in Flight Dynamics and Flight Test Courses.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION	
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS REPORT ☐ DTIC USERS	Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL Richard M. Howard	22b TELEPHONE (*Include Area code*) (408) 646-2870	22c OFFICE SYMBOL AA/Ho

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted; All other editions are obsolete — SECURITY CLASSIFICATION OF THIS PAGE: Unclassified

Approved for public release; distribution is unlimited.

IMPLEMENTATION OF A PERSONAL COMPUTER BASED PARAMETER ESTIMATION PROGRAM

by

Robert Gordon Graham
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1979
M.S., Naval Postgraduate School, 1991

Submitted in partial fulfillment
of the requirements for the degree of

AERONAUTICAL AND ASTRONAUTICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
March, 1992

ABSTRACT

Aircraft parameter estimation is the process of extracting numerical values for aerodynamic stability and control derivatives from flight-test time history data. This process can be used as a verification or validation tool for results obtained from wind-tunnel testing or through computational analysis, and can obtain or improve estimations of dynamic derivatives.

This study implements the MATLAB Personal Computer (PC) based maximum likelihood estimation routine for aircraft longitudinal and lateral-directional derivatives. The parameter estimation was first accomplished on generated simulated data, with and without noise. The noise consisted of measurement and state noise which used the Dryden Gust Model. Secondly, two actual longitudinal flight-test maneuvers are analyzed for the F-14A and the T-37 aircraft. Additionally, the simulated portion of this study can be an excellent instructional aid in Flight Dynamics and Flight Test Courses.

TABLE OF CONTENTS

LIST OF SYMBOLS

A,B,C,D,F,G	System Matrices
A_n	Normal Acceleration
A_y	Lateral Acceleration
b	Wing span, gust model constant
c	Chord length
cg	Center of gravity
C_L	Coefficient of lift
$C_{L_\alpha} = \frac{\partial C_L}{\partial \alpha}$	
$C_{L_{\delta_e}} = \frac{\partial C_L}{\partial \delta_e}$	
C_l	Coefficient of rolling moment
$C_{l_\beta} = \frac{\partial C_l}{\partial \beta}$	
$C_{l_p} = \frac{\partial C_l}{\partial (\frac{pb}{2V})}$	
$C_{l_r} = \frac{\partial C_l}{\partial (\frac{rb}{2V})}$	
$C_{l_{\delta_a}} = \frac{\partial C_l}{\partial \delta_a}$	
$C_{l_{\delta_r}} = \frac{\partial C_l}{\partial \delta_r}$	
C_m	Coefficient of pitching moment
$C_{m_\alpha} = \frac{\partial C_m}{\partial \alpha}$	
$C_{m_q} = \frac{\partial C_m}{\partial (\frac{qc}{2V})}$	

$C_{m_{\delta_e}} = \frac{\partial C_m}{\partial \delta_e}$

C_n	Coefficient of yawing moment

$C_{n_\beta} = \frac{\partial C_n}{\partial \beta}$

$C_{n_p} = \frac{\partial C_n}{\partial (\frac{pb}{2V})}$

$C_{n_r} = \frac{\partial C_n}{\partial (\frac{rb}{2V})}$

$C_{n_{\delta_a}} = \frac{\partial C_n}{\partial \delta_a}$

$C_{n_{\delta_r}} = \frac{\partial C_n}{\partial \delta_r}$

C_y	Coefficient of the Y-force component

$C_{y_\beta} = \frac{\partial C_y}{\partial \beta}$

$C_{y_{\delta_r}} = \frac{\partial C_y}{\partial \delta_r}$

C_z	Coefficient of the Z-force component
dt	incremental time
$\mathbf{F}$	Vector Force
F_x, F_y, F_z	Force components
g	Gravity
$\mathbf{H}, H$	Angular momentum, matrix of unknowns
I	Moment of inertia
$J(\zeta, R)$	Likelihood function
$\mathbf{K}$	Kalman filter gain matrix
k	gust model constant

L, M, N	X, Y, Z moment components
$L_\beta = \bar{q} s b C_{l_\beta}$	
$L_p = \frac{\bar{q} s b^2}{2V} C_{l_p}$	
$L_r = \frac{\bar{q} s b^2}{2V} C_{l_r}$	
$L_{\delta_a} = \bar{q} s b C_{l_{\delta_a}}$	
Lw	Scale of the turbulence
M	Vector Moment
m	Mass
$M_\alpha = \frac{\bar{q} s c}{I_y} C_{m_\alpha}$	
$M_q = \frac{\bar{q} s c^2}{2 I_y V} C_{m_q}$	
$M_{\delta_e} = \frac{\bar{q} s c}{I_y} C_{m_{\delta_e}}$	
$N_\beta = \bar{q} s b C_{n_\beta}$	
$N_p = \frac{\bar{q} s b^2}{2V} C_{n_p}$	
$N_r = \frac{\bar{q} s b^2}{2V} C_{n_r}$	
$N_{\delta_a} = \bar{q} s b C_{n_{\delta_a}}$	
$N_{\delta_r} = \bar{q} s b C_{n_{\delta_r}}$	
p	Roll rate
q	Pitch rate
$\bar{q} = \frac{1}{2} \rho V^2$	
R	Covariance matrix
r	Yaw rate

s	Reference area
t	Time
u, v, w	X, Y, Z velocity components
$\mathbf{V}$	Vector velocity
V	Magnitude of velocity vector
w_g	Z velocity due to gust
X_{an}	x-distance to normal acclerometer
X_{ap}	x-distance to AOA probe
X_{ay}	x-distance to lateral accelerometer
X_{bp}	x-distance to Beta probe

$$Y_\beta = \frac{\bar{q}s}{m} C_{y_\beta}$$

$$Y_{\delta_r} = \frac{\bar{q}s}{m} C_{y_{\delta_r}}$$

Z_{an}	z-distance to normal accelerometer
Z_{ay}	z-distance to lateral accelerometer

$$Z_\alpha = \frac{-\bar{q}s}{m} C_{L_\alpha}$$

$$Z_{\delta_e} = \frac{-\bar{q}s}{m} C_{L_{\delta_e}}$$

α	Angle of Attack (AOA)
β	Side slip angle (Beta)
$\delta_{a,e,r}$	Change in aileron, elevator, rudder
ζ	Set of unknowns
η	Zero mean, white noise
Θ	Pitch angle
λ	Gust model constant, unknowns
ρ	Density
σ_w	rms value of the turbulence

Φ	Roll or bank angle, transition matrix
ψ	Yaw angle, integral of the transition matrix
ω	Angular velocity

Superscripts

^{T}	Matrix transpose
$\dot{}$	First derivative
$\ddot{}$	Second derivative
$\sim$	Predicted
$\hat{}$	Corrected
$\rightarrow$	Vector

Subscript

$_o$	initial value

ACKNOWLEDGEMENTS

There have been numerous people who have influenced and shaped my life. Certainly my parents and grandparents were the predominant individuals shaping me when I was young. One thing that always stood out as significant during my upbringing was the importance of an education. For when we stop learning the end is near. I want to thank those who brought me up emphasizing the importance of continuing ones' education.

I want to also express my thanks to the following people who may not even recognize they contributed: Capt. Mark Erickson, USAF, Major Dan Gleason, USAF, Mr. Scott Hoffman, Mr. Dennis Mar, Mrs. Jan Kleinschmidt, Lt. Mark Whittle, Dr. Louis Schmidt, and Dr. Edward Wu.

Of course, special thanks to my advisor, Professor Rick Howard, who kept my hopes up during the low times and provided the necessary guidance throughout.

Finally, as everyone should know behind each of my successes there was a loving spouse providing the much needed support. Maria, without you none of this would have been possible. This is for you and our boys, Andrew, Nathan, Bobby, and Christopher.

I. INTRODUCTION

Aircraft parameter estimation is the process of extracting numerical values for aerodynamic stability and control derivatives from flight-test time history data. Aircraft flight tests designed for this purpose are generally motivated by one or a combination of the following objectives:

1. The desire to correlate flight test parameter estimates with wind tunnel data and analytic results.
2. The desire to more accurately refine parameter estimates for purposes of control system analysis and design.
3. The desire to achieve an accurate aircraft math model for use in high fidelity flight simulators.

An early and continued use of parameter estimation, as stated above, is in the validation of wind tunnel and analytic results. However, due to the continuing advances in aircraft design and performance capabilities, the ability to accurately extrapolate wind-tunnel test results is diminishing and a greater emphasis is being placed on flight test results. [Ref.1:p.2]

Comprehensive wind-tunnel testing, combined with analytic analysis, can give reasonable estimates of an aircraft's aerodynamic derivatives, but there are potential sources for inaccurate predictions: the matching of "scaled" wind-tunnel tests with expected flight conditions is

difficult, with Reynolds number differences often being the standard explanation for discrepancies. Reliable and accurate dynamic wind tunnel test results are extremely difficult and expensive to accomplish. Support systems (stings, etc.) have become an issue as to the extent the data, especially drag, are affected. [Ref. 2:p.3] Additionally, the ever present time and money constraints often necessitate shortcuts in not only wind-tunnel testing but in flight testing as well. It seems wise, therefore, to use flight-test data, at the very least, as a verification tool of aircraft stability and control derivatives for even the most simple configurations.

Currently at the Naval Postgraduate School (NPS), in the Department of Aeronautics and Astronautics, research is being done using remotely piloted aircraft as research testbeds. One testbed in use has been a half-scale Pioneer Unmanned Air Vehicle (UAV).

The full-scale Pioneer is operational in the U.S. Navy and Marine Corps and saw extensive action in the recent war with Iraq. The small size of the Pioneer or essentially any tactical UAV allows it to operate close to and in some cases behind enemy lines, extending the "eyes" of battlefield commanders. Its missions include gun fire spotting, real time enemy surveillance, bomb damage assessment, target designation and an array of intelligence collection techniques.

The relatively low cost, small size, reduced risk and inherent flexibility of UAV's, such as the half-scale Pioneer,

have allowed the department to become actively involved in assessing their flight characteristics. The Pacific Missile Test Center (PMTC) at Pt. Mugu California is a development and testing facility for the U.S. Navy. Current activity at PMTC includes developmental work in conjunction with the Pioneer UAV. In development by the Target Simulation Lab at PMTC is a flight training simulator to be used by Pioneer operators for initial training and proficiency flights. Results from thesis work of two former NPS students, USMC Capts. Daniel Lyons and Robert Bray, have been supplied to the Lab [Refs. 3 and 4]. These results comprised various aerodynamic parameters obtained from two different approaches, a numerical method (low-order panel technique) and wind tunnel tests of a 0.4-scale model. The aerodynamic data supplied to PMTC are being used in their math model for the simulator.

The goal of the present study is to incorporate a personal computer (PC) parameter estimation capability into the ongoing NPS flight research. This application will give the flight test program an added dimension: the ability to compare data from wind tunnel and analytic analyses with flight test results. Additionally, the adapted program can be incorporated into flight test and dynamic stability and control courses as a valuable teaching aid.

In the near term, interest in the Pioneer parameter estimation results has been expressed by PMTC in hopes of achieving a more realistic training simulator for the

operators. It is hoped that full scale time history data can be obtained from PMTC. Future work in this area includes completion of the Pioneer flight research and comparison of the derivative results obtained from time history parameter estimation with those obtained in References 3 and 4. Additionally, other UAV's in procurement by the Department of the Defense (DOD) could be studied at NPS.

II. BACKGROUND

A. HISTORICAL DEVELOPMENT

The history of flight testing would in itself make an alluring and fascinating book; this cursory summary of parameter estimation and flight testing reviews but a small fraction of the significant events in the history of flight testing. The majority of the historical content was found in the opening remarks given by Herman A. Rediess [Ref. 5] at a 1973 symposium on parameter estimation techniques.

One of the first test programs to obtain quantitative measurements of aircraft aerodynamic characteristics in flight was reported by Warner and Norton in 1919 [Ref. 6]. Tests were conducted on two Curtiss "Jenny" JN-4H biplanes at Langley Field, Virginia. Lift and drag coefficients were estimated by measuring airspeed, pitch attitude and engine speed in flight and assuming certain engine thrust characteristics. This specific flight test was a meager beginning for in-flight testing, but today it is very apparent that in-flight testing is a vital requirement. A 1933 report by Soulé and Wheatly [Ref. 7] is thought to be the first report to have determined and compared major longitudinal stability and control derivatives from flight test data with results acquired through theoretical predictions. The

airplane was the single engine Doyle O-2. The analysis used simplified models, solving for one parameter at a time while assuming values for the other parameters based upon wind tunnel data or other flight tests. Early in the 1950's a major advancement in parameter estimation was achieved by Shinbrot [Ref. 8] using least squares curve fits between the equations of motion and flight data. A considerable drawback at that time was the extensive calculations required for this approach. These calculations, of course, were completed entirely by hand as the digital computer was not yet available as an engineering tool.

Significant improvements were further realized in the later 1950's, and throughout the 60's and 70's, due to:

1. The availability of the digital computer.
2. The progress in the technical disciplines of system identification and numerical analysis.
3. The availability of high speed automatic data acquisition systems.

Again, an excellent overview of the evolution of parameter estimation techniques up to approximately 1970 is contained in Reference 5.

There are numerous methods for extracting the stability and control derivatives from flight-test data that have been developed and tested since the early 50's. Each starts with equations of motion and essentially attempts to curve fit calculated results to the flight-test data by adjusting each

of the derivatives or coefficients in the math model. Some, but certainly not all, techniques that have been used with success include: Ordinary Least Squares, Weighted Least Squares, Deterministic Least Squares, Maximum Likelihood, Statistical Linearized Filter, Extended Kalman Filter, frequency domain methods, and an older technique used in the 1950's, called analog matching. This last technique was a manual curve fitting method using an analog computer in which the results were very much dependent upon the skill of the operator.

B. MODERN DEVELOPMENT

Major contributions to aircraft parameter estimation since the mid 1960's have been made at two NASA facilities, the NASA Ames Research Center's Dryden Flight Research Facility and the NASA Langley Research Center. The parameter estimation contributions from these facilities have been made primarily through the work of Lawrence Taylor, Kenneth Iliff, Richard Maine and James Murray.

Taylor and Iliff developed a Newton-Raphson parameter estimation program in 1966 based upon the theoretical work of Balakrishnan [Ref. 9]. The program used a modified Newton-Raphson algorithm to effect the maximum likelihood technique for estimating stability and control derivatives. This program underwent a gradual evolution during its application [Ref. 10]. The outcome in 1973 was a program named MMLE

(Modified Maximum Likelihood Estimator) which used the same basic algorithm (Newton-Raphson) but incorporated features useful for processing large amounts of flight data. This program was widely circulated among industry and government agencies and as of 1980 had been used to analyze over 35 different aircraft at the NASA Dryden Flight Research Center alone. [Ref. 11:p.2]

Development of MMLE3 (Modified Maximum Likelihood Estimation program version 3) was completed by Maine and Iliff in 1980 in response to a requirement for a more versatile parameter estimation program. MMLE3 had two advances over MMLE: more flexibility in defining the equations of motion (although still linear); and the capability for estimation in the presence of state noise, also called process or input noise, a good example being atmospheric turbulence. [Ref. 11:p. 2] Further details on MMLE3 will be addressed later.

In 1987, a new parameter estimation program to accommodate nonlinear models was developed at NASA Dryden by Maine and Murray [Ref. 12]. This parameter estimation program, named pEst, supports nonlinear models, and thus aircraft dynamic behavior can be tested at extreme flight conditions (high AOA) and for unique configurations (oblique wing, etc.).[Ref. 2]

The basic concepts of aircraft parameter estimation techniques have remained unchanged for over two decades and are shown in Figure 1 [Ref. 5:p.14]. These concepts include:

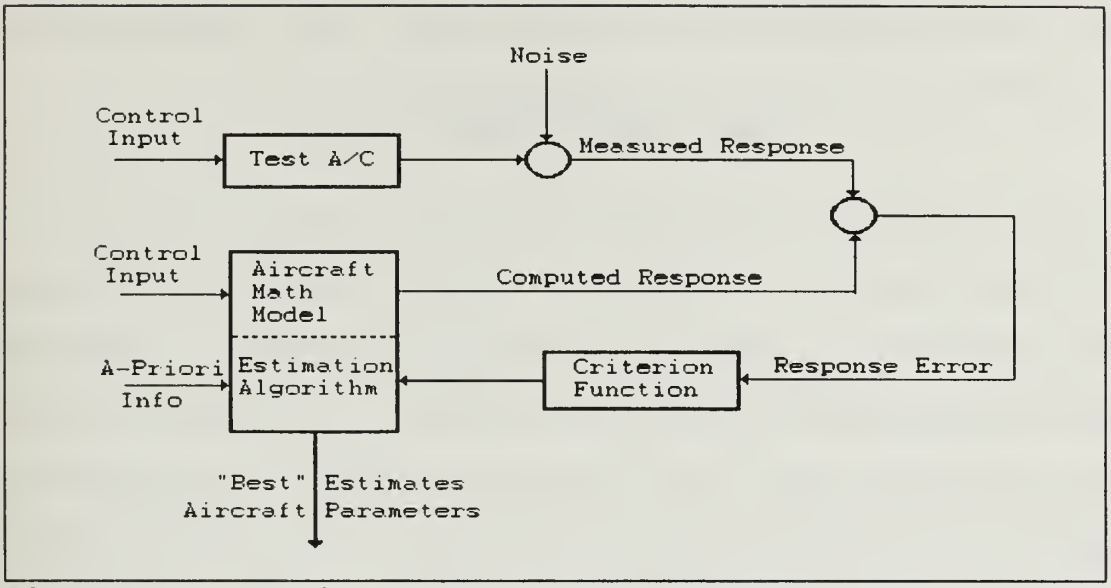

Figure 1. Basic Concepts of Contemporary Parameter Estimation Techniques

(1) the mathematical model, (2) the data acquisition system, (3) the estimation algorithm, and (4) the required test inputs. Each of these elements will be discussed later in more detail.

A Maximum Likelihood (ML) estimator was chosen as our parameter estimator for the following reasons. First, this method has become and still is "accepted as the standard approach to determining aircraft stability and control derivatives from flight data" [Ref.13:p.558]. Furthermore, the flight regimes of the test vehicles at NPS, at least initially, are expected to be well within the region where a linear math model will provide accurate parameter estimations. Furthermore, a PC compatible ML estimator program was chosen because of the PC's flexibility and availability at NPS. This combination appeared ideal for use in analyzing the ongoing

NPS flight research and also for use in the classroom as an enhancement to existing teaching aids.

III. MATHEMATICAL MODEL

In this section, the linear equations of motion used to describe a typical manned or unmanned aircraft will be developed. Similar developments are shown in most aircraft dynamics textbooks. References used in this development were *Airplane Flight Dynamics* by Roskam [Ref. 14], classroom notes by NPS Professor Louis Schmidt [Ref. 15], the textbook *Flight Stability and Automatic Control* by Nelson [Ref. 16] and Reference 2. In creating simulated data a gust or turbulence model was used and that too is developed in this section. Lastly, a discussion of the observation equation corrections is presented.

A. MODEL EQUATION DEVELOPMENT

This development begins with Newton's linear and angular momentum equations:

$$\vec{F}=\frac{d}{dt}(m\vec{V}) \tag{1a}$$

$$\vec{M}=\frac{d}{dt}(\vec{H}) \tag{1b}$$

Where the force, **F**, is the sum of the externally applied forces and the moment, **M**, is the sum of the applied moments about the center of gravity (cg). The use of non-rotating, earth reference coordinates for equations 1a and 1b is

unwieldy for two reasons. First, required measurements are predominately made in the rotating body axis system; and secondly, but of more significance, the inertia matrix or tensor is a function of time in the non-rotating system. Therefore, the axis system chosen is the standard body axis system, shown in Figure 2 [Ref. 15]. The body axis system is used by Iliff and Maine in Reference 2, but the reader should be advised that the equations of motion are also at times derived using the stability axis system. Reference 14 has a good description of the differences between the two axis systems. The origin is positioned at the vehicle's cg with the X-direction pointing out the nose of the aircraft, the Y-direction out the starboard wing and the Z-direction out the bottom of the aircraft. Transforming equations 1a and 1b into the rotating body axis system is done below:

$$\vec{F}=\frac{\partial}{\partial t}(m\vec{V})+\vec{\omega}\times(m\vec{V}) \tag{2a}$$

$$\vec{M}=\frac{\partial}{\partial t}(\vec{H})+\vec{\omega}\times\vec{H} \tag{2b}$$

where the angular momentum, **H**, is the inner dot product of the aircraft mass moment of inertia matrix, $[\mathbf{I}_{ac}]$, with the angular rotation vector ω. The inertia matrix in the body axis coordinate system is *not* a function of time as it is in the earth reference. The above equations are vector equations and can be written into scalar components.

The components in the body axis for the ω vector are p, q, and r for the roll, pitch and yaw rates, respectively. The components in the body axis for **V** are u, v, and w for the X, Y, and Z components of velocity, also shown in Figure 2.

The applied forces on the aircraft can be broken down into aerodynamic, gravitational and thrust components. (The thrust

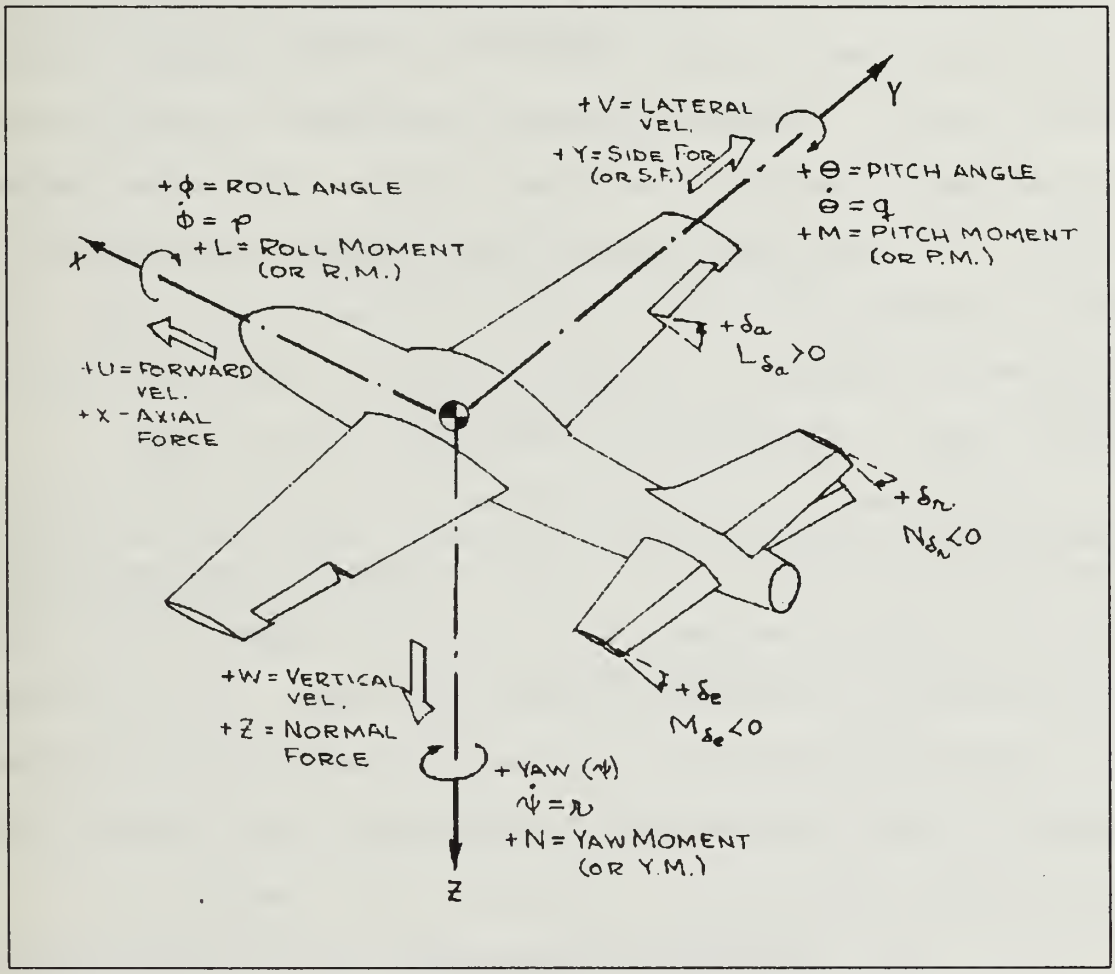

Figure 2. Body Axis System and Notation

is assumed to act along the X-body axis.)

The moments can be broken down into just aerodynamic components, since the thrust is assumed, and gravity forces,

by definition, act through the cg, and their moment contributions are zero. The moment components are then due to the aerodynamic forces and are shown in Figure 2 as **L**, **M**, and **N**.

Since the gravity force components in the body axis system depend on the orientation of the airplane relative to earth-fixed coordinates (assuming a flat non-rotating earth), it is necessary to describe the orientation of the aircraft relative to the earth. Euler angles are introduced to accomplish this transformation between coordinate systems. The Euler angles, Ψ, Θ, and Φ, are three consecutive rotation angles needed for the transformations from one axis system to the other. The angle Ψ is referred to as the yaw angle. The angle Θ is the pitch angle. The angle Φ is the bank or roll angle.

The aircraft is further assumed to be rigid; that is, the mass particles remain at constant distances from each other. The X-Z plane is assumed to be a plane of symmetry and thus, the products of inertia I_{xy} and I_{yz} are zero. In this model, the rotating engine parts and sloshing fuel are being ignored, and over short periods of time when data are collected, the mass of the aircraft is considered to be constant.

The force component equations that are derived with the above assumptions are:

$$F_x = m(\dot{u} + qw + rv) \tag{3a}$$

$$F_y = m(\dot{v} + ru + pw) \tag{3b}$$

$$F_z = m(\dot{w} + pv + qu) \tag{3c}$$

and the moment component equations are:

$$L = M_x = \dot{p}I_x - \dot{r}I_{xz} + qr(I_z - I_y) - pqI_{xz} \tag{4a}$$

$$M = M_y = \dot{q}I_y + rp(I_x - I_z) + (p^2 - r^2)I_{xz} \tag{4b}$$

$$N = M_z = -\dot{p}I_{xz} + \dot{r}I_z + pq(I_y - I_x) + qrI_{xz} \tag{4c}$$

Where the left hand sides of the above force and moment component equations are:

$$F_x = \bar{q}sC_x - mg\sin\theta + Thrust \tag{5a}$$

$$F_y = \bar{q}sC_y + mg\sin\phi\cos\theta \tag{5b}$$

$$F_z = \bar{q}sC_z + mg\cos\phi\cos\theta \tag{5c}$$

$$M_x = \bar{q}sbC_l \tag{6a}$$

$$M_y = \bar{q} s c C_m \tag{6b}$$

$$M_z = \bar{q} s b C_n \tag{6c}$$

where $\bar{q}$ is the dynamic pressure and q is the pitch rate.

The vehicle is assumed to operate at small side slip angles and small perturbations around a steady state condition. The perturbations are assumed to be small in order that the sines and cosines of the disturbance angles are approximately the angles themselves and one (1) respectively, and the products and squares of the products are negligible when compared to the quantities themselves. This approximation is termed small perturbation theory and permits the equations of motion to be decoupled and linearized into two smaller subsets: Lateral-Directional and Longitudinal.

It is often times more convenient to have the equations in terms of α (angle of attack), β (sideslip angle), and V than in terms of u, v, and w. These angles are usually measured directly vice the velocity components. In the transformation of the equations of motion into equations with α and β, the following can be noted:

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \tag{7a}$$

$$\beta = \tan^{-1}\left(\frac{v}{u}\right) \tag{7b}$$

and if the angles are small, then

$$\alpha \approx \frac{w}{u} \tag{8a}$$

$$\dot{\alpha} \approx \frac{\dot{w}}{u} \tag{8b}$$

$$\beta \approx \frac{v}{u} \tag{8c}$$

$$\beta \approx \frac{\dot{v}}{u} \tag{8d}$$

These equations will be used in the next sections to construct the basic aircraft model.

B. SIMPLIFIED LONGITUDINAL EQUATIONS

The longitudinal set of equations pertains to rotation or moments about the Y axis (4b and 6b) with translation or forces along the X and Z axis (3a, 3c, 5a and 5c). With the substitution of 8a and 8b from above, and also with the use of

small perturbation theory, the following simplified longitudinal equations are formulated:

$$\dot{\alpha} = \frac{\bar{q}s}{mu} C_z + q + \frac{g}{u} \tag{9}$$

$$\dot{q} = \frac{\bar{q}sc}{I_y} C_m \tag{10}$$

$$\theta = q \tag{11}$$

where the lift is approximately parallel to the Z-axis, so

$$C_L \approx -C_z \tag{12a}$$

$$C_L = C_{L_\alpha} \alpha + C_{L_{\delta_e}} \delta_e + C_{L_o} \tag{12b}$$

$$C_m = C_{m_\alpha} \alpha + C_{m_q} \frac{qc}{2V} + C_{m_{\delta_e}} \delta_e + C_{m_o} \tag{12c}$$

Equations 12a and 12b are then substituted into equation 9 while equation 12c is substituted into equation 10. The result expressed in state-space format and using dimensional derivatives follows:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{Z_\alpha}{V} & 1 & 0 \\ M_\alpha & M_q & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} \alpha \\ q \\ \theta \end{pmatrix} + \begin{bmatrix} \frac{Z_{\delta_e}}{V} & \frac{g}{V} \\ M_{\delta_e} & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} \delta_e \\ 1 \end{pmatrix} \tag{13a}$$

with the output equation being:

$$\begin{bmatrix} \alpha \\ q \\ \theta \\ A_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \frac{Z_\alpha}{g} & 0 & 0 \end{bmatrix} \begin{pmatrix} \alpha \\ q \\ \theta \end{pmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{Z_{\delta_e}}{g} & A_{n_o} \end{bmatrix} \begin{pmatrix} \delta_e \\ 1 \end{pmatrix} \tag{13b}$$

In equation 13a the M_q term includes the $M_{\dot{\alpha}}$ term. The measured variables are V, α, q, θ, A_n (normal acceleration in "g's") and δ_e during the maneuvers. The magnitude of the velocity, V, is approximately equal to u at small α and has been substituted for u in the above state space equation. The dimensional derivatives formulation is shown in the list of symbols.

C. SIMPLIFIED LATERAL-DIRECTIONAL EQUATIONS

The lateral-directional equation set pertains to the rotation about the X and Z axes (4a, 4c, 6a and 6c) and translation along the Y axis (3b and 5b). These equations are used to derive the lateral-directional state space representation. Small perturbation theory, and the use of equations 8c and 8d, are used to obtain the following lateral-directional equations:

$$C_y = C_{y_\beta}\beta + C_{y_{\delta_r}}\delta_r \tag{14a}$$

$$\beta = \frac{\bar{q}sC_y}{mu} + \frac{g\phi}{u} - r \tag{14b}$$

$$C_l = C_{l_\beta}\beta + C_{l_p}\frac{pb}{2V} + C_{l_r}\frac{rb}{2V} + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r \tag{15a}$$

$$\dot{p}I_x - \dot{r}I_{xz} = \bar{q}sbC_l + qr(I_y - I_z) + pqI_{xz} \tag{15b}$$

$$C_n = C_{n_\beta}\beta + C_{n_p}\frac{pb}{2V} + C_{n_r}\frac{rb}{2V} + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r \tag{16a}$$

$$-\dot{p}I_{xz} + \dot{r}I_z = \bar{q}sbC_n + pq(I_x - I_y) - qrI_{xz} \tag{16b}$$

and

$$\frac{d(\phi)}{dt} = \dot{\phi} = p + r\tan\theta\sin\phi + q\tan\theta\cos\phi \tag{17}$$

Substituting equations 14a, 15a, and 16a into 14b, 15b and 16b respectively, and again expressing these equations along with equation 17 in state-space format using dimensional derivatives the following is obtained:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & I_x & -I_{xz} & 0 \\ 0 & -I_{xz} & I_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \end{pmatrix} = \begin{bmatrix} \frac{Y_\beta}{V} & 0 & -1 & \frac{g}{V} \\ L_\beta & L_p & L_r & 0 \\ N_\beta & N_p & N_r & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{pmatrix} \beta \\ p \\ r \\ \phi \end{pmatrix} + \begin{bmatrix} 0 & \frac{Y_{\delta_r}}{V} & 0 \\ L_{\delta_a} & L_{\delta_r} & 0 \\ N_{\delta_a} & N_{\delta_r} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \delta_a \\ \delta_r \\ 1 \end{pmatrix} \quad (18a)$$

and output equation:

$$\begin{pmatrix} \beta \\ p \\ r \\ \phi \\ A_y \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{Y_\beta}{g} & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \beta \\ p \\ r \\ \phi \end{pmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{Y_{\delta_r}}{g} & A_{y_o} \end{bmatrix} \begin{pmatrix} \delta_a \\ \delta_r \\ 1 \end{pmatrix} \quad (18b)$$

The measured variable are the δ_a and δ_r control deflections, β, p, r, Φ, A_y and V.

The two state space representations (longitudinal and lateral-directional) form the mathematical model used with the parameter estimation program to estimate the stability and control derivatives.

D. TURBULENCE MODEL

In preparation for using actual time history data, simulated longitudinal and lateral-directional data were created. In creating the simulated data it was desired to match the expected phenomena or real world effects that would be encountered. Thus turbulence (state noise) and measurement noise were selectively added to the simulated data by the

user. This section describes the model used to generate the turbulence or state noise.

The development begins with the application of the Dryden gust model [Refs. 15 and 18]. The turbulent effect can be added to both the longitudinal and lateral-directional equations. The development for the longitudinal case will be shown below and can be extrapolated in a straightforward manner for the lateral-directional case.

In the Laplace domain the vertical gust velocity transfer function is

$$w_g(s) = \sqrt{k}\frac{s+b}{(s+\lambda)^2}\eta(s) \tag{19}$$

where

$$b = \frac{u}{\sqrt{3}L_w} \tag{20a}$$

$$\lambda = \frac{u}{L_w} \tag{20b}$$

$$k = \frac{3\sigma_w^2 u}{\pi L_w} \tag{20c}$$

and L_w is the scale of the turbulence, σ_w is the rms value of the turbulence and η is a zero mean white noise input. The values used for L_w and σ_w equated to a turbulence level between light and moderate, and can be adjusted if desired.

Equation 21 was obtained by transforming equation 19 from the Laplace domain into the time domain and dividing by u:

$$\frac{w_g}{u} = \ddot{\alpha}_g + 2\lambda\dot{\alpha}_g + \lambda^2\alpha_g = \frac{\sqrt{k}}{u}(b\eta + \dot{\eta}) \tag{21}$$

where α_g is the α perturbation attributable to the gust.

Equation 21 is converted into the state space format by letting

$$Z_1 = \alpha_g \tag{22a}$$

$$Z_2 = \dot{\alpha}_g - \frac{\sqrt{k}}{u}\eta \tag{22b}$$

$$\dot{Z}_1 = Z_2 + \frac{\sqrt{k}}{u}\eta \tag{22c}$$

$$\dot{Z}_2 = -2\lambda Z_2 - \lambda^2 Z_1 + \frac{\sqrt{k}}{u}\eta(b - 2\lambda) \tag{22d}$$

it follows that

$$\begin{bmatrix} \dot{\alpha}_g \\ \ddot{\alpha}_g \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{bmatrix} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} + \begin{bmatrix} \frac{\sqrt{k}}{u} \\ \frac{\sqrt{k}}{u}(b-2\lambda) \end{bmatrix} (\eta) \qquad (23)$$

This result can be combined with the longitudinal state space equation 13 (Lateral-Directional equation 18) to yield equation 24a:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \\ \dot{Z}_1 \\ \dot{Z}_2 \end{bmatrix} = \begin{bmatrix} \frac{Z_\alpha}{V} & 1 & 0 & \frac{Z_\alpha}{V} & 0 \\ M_\alpha & M_q & 0 & M_\alpha & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\lambda^2 & -2\lambda \end{bmatrix} \begin{pmatrix} \alpha \\ q \\ \theta \\ Z_1 \\ Z_2 \end{pmatrix} + \begin{bmatrix} \frac{Z_{\delta_e}}{V} & 0 & \frac{g}{V} \\ M_{\delta_e} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{\sqrt{k}}{V} & 0 \\ 0 & \frac{\sqrt{k}}{V}(b-2\lambda) & 0 \end{bmatrix} \begin{pmatrix} \delta_e \\ \eta \\ 1 \end{pmatrix} \qquad (24a)$$

and the output equation 24b:

$$\begin{bmatrix} \alpha \\ q \\ \theta \\ A_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \frac{Z_\alpha}{g} & 0 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \alpha \\ q \\ \theta \\ z_1 \\ z_2 \end{pmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{Z_{\delta_e}}{g} & 0 & A_{n_o} \end{bmatrix} \begin{pmatrix} \delta_e \\ \eta \\ 1 \end{pmatrix} \qquad (24b)$$

Similarly, the state space equations for the lateral-directional model with turbulent noise can be developed and are shown below:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_x & -I_{xz} & 0 & 0 & 0 \\ 0 & -I_{xz} & I_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{Z}_1 \\ \dot{Z}_2 \end{pmatrix} = \begin{bmatrix} \frac{Y_\beta}{V} & 0 & -1 & \frac{g}{V} & \frac{Y_\beta}{V} & 0 \\ L_\beta & L_p & L_r & 0 & L_\beta & 0 \\ N_\beta & N_p & N_r & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -\lambda^2 & -2\lambda \end{bmatrix} \begin{pmatrix} \beta \\ p \\ r \\ \phi \\ Z_1 \\ Z_2 \end{pmatrix} \quad \cdots\cdots$$

$$+ \begin{bmatrix} 0 & \frac{Y_{\delta_r}}{V} & 0 & 0 \\ L_{\delta_a} & L_{\delta_r} & 0 & 0 \\ N_{\delta_a} & N_{\delta_r} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\sqrt{k}}{V} & 0 \\ 0 & 0 & \frac{\sqrt{k}}{V}(b-2\lambda) & 0 \end{bmatrix} \begin{pmatrix} \delta_a \\ \delta_r \\ \eta \\ 1 \end{pmatrix} \qquad (25a)$$

$$\begin{pmatrix} \beta \\ p \\ r \\ \phi \\ A_y \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{Y_\beta}{g} & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \beta \\ p \\ r \\ \phi \\ Z_1 \\ Z_2 \end{pmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{Y_{\delta_r}}{g} & 0 & A_{y_o} \end{bmatrix} \begin{pmatrix} \delta_a \\ \delta_r \\ 1 \end{pmatrix} \qquad (25b)$$

E. MEASUREMENT CORRECTIONS

The observed or measured data must be corrected for measurement errors caused by the positioning of the sensors. The aircraft equations of motion were developed for the aircraft cg. Therefore, measurements taken by sensors not physically located at the cg require corrections to reference the values to the cg location. This subsection will document

the correction equations used in this analysis. The simulated data need not be corrected since these data were manufactured at the cg of the aircraft. However, in anticipation of actual aircraft flight test parameter estimation, sensor position corrections were implemented into the programs used to analyze genuine flight data.

1. **Longitudinal**

The longitudinal α and A_n data are capable of corrections for sensor position displacement from the cg. The α was corrected for the X-coordinate probe position forward (+) or aft (-) of the cg (X_{ap}) as shown below:

$$\alpha_{cg}=\alpha_{probe}+\frac{X_{ap}}{V}q \tag{26a}$$

A correction for the upwash angle α_w at the probe was not taken into account as it is assumed to be small or previously accounted for when the sensor was calibrated; a more complete discussion on corrections is contained in Reference 2. The normal acceleration, A_n, was corrected for both X_{an} (fwd +) and Z_{an} (down +) displacement from the aircraft cg as shown below:

$$A_{n_{cg}}=A_{n_{acc}}-\frac{X_{an}}{g}\dot{q}-\frac{Z_{an}}{g}q^2 \tag{26b}$$

2. Lateral-Directional

The sideslip angle, β, and the lateral acceleration, A_y, were corrected for their sensor positions and the correction equations are shown below:

$$\beta_{cg}=\beta_{probe}-\frac{X_{bp}}{V}r \tag{27a}$$

and

$$A_{y_{cg}}=A_{y_{acc}}-\frac{X_{ay}}{g}\dot{r}+\frac{Z_{ay}}{g}\dot{p} \tag{27b}$$

The values for X_{bp}, X_{ay} and Z_{ay} are again defined positive forward and down analogous to the body coordinates.

IV. MAXIMUM LIKELIHOOD PARAMETER ESTIMATION

A. THEORY

The concept of parameter estimation, as discussed earlier, can be defined quite simply in general terms. The system, in this case a UAV or some other testbed whose parameters are to be estimated, is assumed to be described by a set of linear dynamic equations, a mathematical model, which was defined earlier. To determine the values for the unknown parameters the system is excited by an input. The input and the system's actual response are measured. The values of the system unknowns are then calculated based on the requirement that the model response to the input match the actual system response. Complications to this simplified explanation arise when the following are considered:

1. Measurement noise - perfect measurements are unattainable with any sensor.
2. State noise - the aircraft or system is being excited by unmeasured sources such as atmospheric turbulence.
3. Modeling errors - exactly describing the physical system with simple, especially linear, dynamic equations is very unlikely.

In fact, if the above complications were not present, the exact values of the unknown parameters could be found and what

is termed parameter "identification" vice "estimation" would be our accomplishment.

The common approach for handling the modeling error is to ignore it and let the error be treated as measurement or state noise or both. Iliff and Maine state that "this procedure is not rigorously justifiable, but combined with a carefully chosen model, it is probably the best approach available." [Ref. 19:p. 2] The Modified Maximum Likelihood Estimation algorithm version 3 (MMLE3) program was structured to take into account the presence of state and measurement noise. The information in the following section is a compilation of information on MMLE parameter estimation from References 2 and 19 through 22.

B. MODIFIED MAXIMUM LIKELIHOOD ESTIMATION

In this section the Modified Maximum Likelihood Estimation algorithm version 3 (MMLE3) is presented. The first step for parameter estimation then, as mentioned above, is to model the system accurately. That model was developed in the previous chapter. The aircraft equations of motion define the system model and can be expressed in the following state space form:

$$x(t_0) = x_0 \tag{28}$$

$$\dot{x}(t) = Ax(t) + Bu(t) + Fw(t) + b_x \tag{29a}$$

$$y(t) = Cx(t) + Du(t) + b_y \tag{29b}$$

$$z(k) = y(k) + Gv(k) \tag{29c}$$

where x(t) is the state vector (x_o being the initial state), u(t) is the control input vector, and y(t) is the prediction or model output vector. Matrices A, B, C, and D contain the unknown system parameters, which in this case are the stability and control derivatives. Matrices F and G represent the covariance matrices of the state and measurement noise respectively. The measured response vector, z(k), is sampled at N discrete time points (k=1,...,N).

The state or process noise, w(t), is assumed to be zero-mean, white Gaussian and stationary. The measurement noise, v(t), is assumed to be zero-mean Gaussian noise with identity covariance.

The complete unknown parameter vector to be estimated is then given by:

$$\zeta = (H^T; \lambda^T; b_x^T; b_y^T) \tag{30}$$

where H represents the unknown parameters in the matrices A, B, C, and D; λ represents the unknown elements of the F matrix; and b_x and b_y represent the unknown biases of the state and model output equations respectively. The $[-]^T$ indicates the transpose of a matrix.

The maximum likelihood estimates are obtained by minimizing the negative logarithm of the likelihood function. The likelihood function, J, is a function of the difference between the measured and computed time histories. The likelihood function is:

$$J(\zeta, R) = \frac{1}{2}\sum_{k=1}^{N} [z(k) - y(k)]^T R^{-1} [z(k) - y(k)] + \frac{N}{2} \ln|R| \qquad (31)$$

where R is the innovation covariance matrix. The innovations or residuals are [z(k)-y(k)]. In order to obtain the predicted state variables it is necessary to use a state estimator. The Kalman filter, which is an optimal linear state estimator, is used for this purpose. The Kalman filter consists of a prediction step and a correction step [Ref. 20:p. 12] for equations 29a and 29b and is shown below:

$$\tilde{x}(k+1) = \Phi \hat{x}(k) + \Psi B \bar{u}(k) + \Psi b_x \qquad (32)$$

$$y(k) = C\tilde{x}(k) + Du(k) + b_y \qquad (33)$$

$$\hat{x}(k) = \tilde{x}(k) + \mathbf{K}[z(k) - y(k)] \qquad (34)$$

$$\bar{u}(k) = \frac{1}{2}[u(k+1) + u(k)] \qquad (35)$$

where ~ (tilde) and ^ (circumflex) denote the predicted and corrected state variables respectively. K represents the Kalman filter gain matrix. The state transition matrix is Φ and its integral is Ψ.

1. Cost Function Minimization

The maximum likelihood estimates for the unknowns are found by minimizing the negative logarithm of the likelihood function, $J(\zeta,R)$. The negative logarithm of the likelihood function is often called or referred to as the cost function. This minimization is done by using a modified Newton-Raphson technique which iterates on the vector of unknowns, ζ, with each iteration providing a new estimate of the unknown vector. These new estimates update the math model coefficients, providing a new calculated response and a new response error. This iteration process is continued until the convergence criterion is satisfied.

"The maximum likelihood estimation method has the desirable characteristics of yielding asymptotically unbiased, consistent and efficient estimates [Ref. 19:p.3]."

2. A-Priori Weighting

The MMLE3 algorithm allows for the use of a weighting function to account for prior 'engineering' knowledge of the aircraft parameters. This prior knowledge can be obtained from other test cases, wind tunnel measurements, or indepth analytic analysis. A table of the relative importance and the

prediction accuracy of stability derivatives using theoretical methods is contained in Reference 14, page 236 and can be used as a guide, if desired, to weighting the initial parameter estimates.

The a-priori information can assist the algorithm in converging, but caution should be used, as the weightings can prejudice the answers toward the analyst's own values [Ref.22:p.ST-8].

3. **Estimate Uncertainty**

The use of the Cramér-Rao bounds with the maximum likelihood estimator can also provide a measure of the relative accuracy of the estimates. Each parameter bound gives an approximation to the standard deviation of the estimates. It is important to recognize that these bounds are, in fact, the lower limits for the standard deviation, meaning that the standard deviation value is at least as large as the Cramér-Rao bounds [Ref. 21:p. 12]. More details on the Cramér-Rao bound is contained in References 2, 20, 21 and 22.

The State Space Identification Toolbox (SSIT) for the 386-MATLAB personal computer program implements the MMLE3 algorithm. MATLAB is a registered trademark for matrix oriented software distributed under license agreement by *The Mathworks, Inc.* Reference 21 is a report of the results of a study comparing the mainframe based MMLE3 program and the MATLAB SSIT implementation of MMLE3. The analysis indicated that the PC version results were "generally well within the uncertainty levels of the mainframe parameter estimates [Ref. 21:p.83]". The use of MATLAB Software will be discussed in the following chapter.

V. APPLICATION

A. DATA ACQUISITION

The data acquisition system is an important part of dynamic stability and control testing. The more information known on the details of the entire data acquisition system, the greater the probability that the test results can be more precise. With few details known about the data, often times only gross characteristics of the aircraft can be determined. The details essential for a complete analysis of the data should include how the data were filtered, digitized, time tagged, transmitted and recorded. The complete analysis of the data acquisition system should start at the beginning with the sensors and continue through to the final recorded data product.

Sensor calibration errors, temperature effects, added noise from aircraft vibration, recorders and transmitters are but a small portion of the circumstances that should be reviewed. Common recording systems, their advantages and disadvantages, plus other issues relevant to the entire data acquisition process are discussed in greater detail in Chapters VII and VIII of Reference 2.

B. INPUT SELECTION

The selection of the control inputs for use in parameter estimation must take into account the pilot's acceptability and safety of flight concerns as well as the model validity considerations.

References 1, 2, 22 and 23 detail various methods for the input design employed in aircraft parameter estimation.

One specific requirement is that the controls applicable to the specific model need to be exercised, such that the aircraft modes are excited. Control inputs which are near the frequency of the excited mode usually provide the best results. This is because at these modal input frequencies, the largest aircraft response for a given input usually occurs and provide the estimator significant data as compared to the noise (gust and measurement noise). Judgment must be used when selecting the control inputs to insure flight safety and to avoid responses that exceed any preset magnitude restrictions. In the assumed linear model, for example, as the response magnitudes exceeded the small perturbation assumptions (10-15 degrees), the final parameter estimation results can be expected to worsen. Likewise, the signal-to-noise ratio of the data improves proportionally with the magnitude of the response; thus there is a trade-off in the development of the aircraft control inputs. Reference 23 discusses a method to optimize the control inputs while

accounting for specific restrictions or trade-offs as mentioned above.

The control inputs used to generate the simulated data for this study were elevator, rudder and aileron ramped doublets. This type of control input was selected for two reasons. First, this input is truly representative of actual pilot inputs (impulses and step inputs are not physically realizable), and second, it can be easily adjusted in magnitude and frequency as necessary for different aircraft. The previously mentioned references provide additional information on the specifics of designing control inputs.

C. MATLAB

MATLAB is a commercially available software package for scientific and engineering applications. The program integrates numerical analysis, matrix computation and graphics into a relatively simple environment without the need for traditional programming knowledge. MATLAB has specialized toolboxes for added capabilities. In this study the Control Systems Toolbox and the State-Space Identification Toolbox (SSIT) in addition to the main 386-MATLAB program were used. The SSIT implements the MMLE3 algorithm, discussed previously in Chapter IV.

Again, the use of the MATLAB based program is desirable at NPS because it operates in a familiar PC environment, the importation of data is relatively straightforward and easily

accomplished, knowledge of a formal programming language is not required, and the plotting and hard copy functions are easy to use and manipulate. Furthermore, during the required basic dynamics and linear systems courses, students at NPS use MATLAB extensively as an instructional and problem-solving aid.

1. M-files

MATLAB is capable of executing sequences of commands stored in files, called M-files or macros, from a single-line command. The M-files have a file type of .m and consist of a sequence of normal MATLAB statements that can include the execution of other M-files. Major benefits of the M-files are: repetitive or long sequences of commands can be automated; new functions can be created by the user for a specific need; and the .m files are ASCII type format and easily edited.

The following sections will describe the M-files created during this study for use in implementing the PC MATLAB parameter estimation program. It is noteworthy that the SSIT is itself an M-file (mmle.m). More detailed information concerning MATLAB is contained in the MATLAB user's guides References 22 and 24.

a. Simulated Data

Simulated time history data were created using MATLAB M-files. This simulation was done in preparation for

using actual aircraft flight data in the parameter estimation program. The aircraft models (longitudinal and lateral-directional) used to create the data were previously discussed, as was the turbulence model. The M-files described in the following sections are contained in Appendix A.

(1) Longitudinal

The simulated longitudinal data were created using the M-file LONGDAT.M (see Appendix A). This M-file is extensively commented which allows the user to understand the program and change or adjust certain parameters as necessary, such as turbulence level, measurement noise level, the elevator input amplitude, and period, or to design a completely new input.

Aircraft derivatives and other physical data are necessary to create the simulated data. These data are stored into MATLAB data files for a small number of specific aircraft. These data files are .mat type files. The user can select one of these aircraft or input the data for an aircraft of his or her choosing. If a new aircraft is selected, the data required by the program are interactively requested using input commands. The data are then stored and available in a .mat data file for later use. The storing of these data into accessible files eliminates the need to reenter the data every time additional simulated data are desired for the same type aircraft. Aircraft data from

Reference 16 were initially entered for the following aircraft: NAVION, A4D, F104A, JETSTAR, and B747. The Pioneer UAV data were also entered and were obtained from Reference 4. The general input requirements for the LONGDAT.M macro are:

1. The aircraft physical data, if not using an aircraft with previously saved data

2. The selection of either adding or not adding state and measurement noise to the data

3. The flight specifics: velocity, pressure altitude, and outside air temperature

The M-file uses a ramped elevator doublet for the input to the aircraft math model, which then produces the output. The simulation can be done either with or without noise. When noise is selected, in addition to the state turbulence noise added, a uniform measurement noise is also added to the outputs. The measurement noise added to the outputs α and Θ is zero mean, ±½ degree maximum value, while the measurement noise added to the output q is zero mean, ±2.5 degrees per second maximum value and the measurement noise added to the normal acceleration, A_n, is zero mean, ±.01 g maximum value.

The final output time histories for the user include δ_e, α, q, Θ and A_n plots displayed on the PC monitor, with data files saved containing the simulated time history data. The data files that are created by LONGDAT.M are then

available for the parameter estimation M-files, which will be discussed later.

(2) Lateral-Directional

The simulated lateral-directional data were created using the macro LATDIR.M. This file is similar to LONGDAT.M but uses the lateral-directional math model. The model has two inputs, δ_r and δ_a. The necessary constants and stability and control derivatives again are stored into a .mat file as was done for the longitudinal case. The noise is essentially the same as in the longitudinal model except that the turbulent gusts are caused by a perturbation in the side slip angle, β_g. The uniform measurement noise that is added to the output angles (β and Φ) is zero mean, ±½ degree maximum value, while the noise added to the output angular rates (p and r) is zero mean, ±2.5 degrees per second maximum value, and the noise added to the lateral acceleration is the same as in the longitudinal case.

The outputs are time history plots and data files of rudder and aileron inputs, δ_r and δ_a; side slip, β; roll rate, p; yaw rate, r; roll angle, Φ; and lateral acceleration, A_y. The data files that are created by LATDIR.M are now available for the parameter estimation M-files, which will be discussed next.

b. Parameter Estimation

Once the required time history data are available, either created by simulation or acquired from actual flight tests, the parameter estimation algorithm is used to calculate the 'best' estimate of the stability and control derivatives. The process of executing the MATLAB SSIT parameter estimation program was accomplished with four basic M-files. The four M-files were developed so the execution of the SSIT M-file (mmle.m) would appear transparent to the user. Modifications to the basic four M-files were necessary for each of the following cases:

1. Simulated longitudinal data
2. Simulated lateral-directional data
3. Actual longitudinal flight data
4. Actual lateral-directional flight data

These M-files are included in Appendix A. The four basic M-files used with the simulated data will be discussed followed by the changes needed for the actual flight data cases.

The four basic M-files used in each of the above four cases were designed to simplify the execution of the SSIT M-file (mmle.m). This M-file arrangement is shown by the block diagram in Figure 3 below:

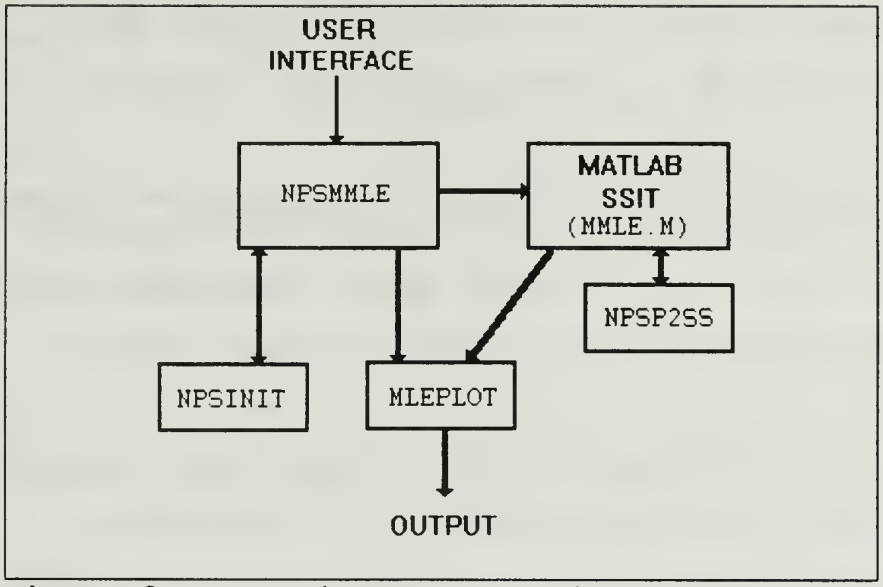

Figure 3 Block Diagram of M-file Arrangement

The four cases mentioned above each have the four basic M-files shown in the above block diagram. The above case number is included in the M-file name used for that case to distinguish the files between the different files. Four files for each of the four cases equate to 16 different M-files. Each set of files is a variation of the basic four M-files.

The only M-file the user initiates is the macro named NPSMMLE_ (the _ is where the case number from above is included with the M-file name) shown in Figure 3. The other macros are "called" in a manner similar to that for a subroutine call in FORTRAN programming.

NPSMMLE_ is initiated by the user and it acts as the link between the SSIT, the other macros and the user. The one exception is that the time between data points, dt, must

be edited in NPSP2SS_.M for cases 3 and 4. The functions of NPSMMLE_ include the following:

1. It initiates the MATLAB diary function which saves a copy of the output (no plots) for subsequent printing

2. It "calls" the NPSINIT_ macro (discussed later)

3. It arranges the time history data into the required format

4. It establishes and makes known the function file to the SSIT, which converts the parameter vector into the user defined state space description of the model

5. It defines additional information to the SSIT which is discussed in detail in References 21 and 22

6. It "calls" the SSIT parameter estimation program (mmle.m)

7. It formats and displays the final numerical results to the monitor

8. It "calls" the MLEPLOT_ macro which graphically displays the results

The NPSINIT_ macro is "called" by the NPSMMLE_ macro and performs the following functions:

1. It loads the time history data file supplied by the user. This data file is either obtained by simulation or from actual flights.

2. It requests the time interval between data points.

3. It requests the vehicle's physical attributes, initial parameter estimates and the flight conditions.

All of the above information is saved and then made available for the SSIT when it is reloaded into the MATLAB working environment by NPSMMLE_.

The NPSP2SS_ macro is a function file used by SSIT during the parameter estimation process. The file converts the parameter vector into the user defined stated space description of the system. Details of the P2SS M-file are contained in References 21 and 22.

After the SSIT has been "called" and successfully run, the numerical results are output to the analyst by way of the PC monitor. The MLEPLOT_ macro is then initiated and its function is to display the results graphically to the user. These plots are shown on the monitor and saved as MATLAB .MET files. These .MET files are high resolution graphics files that may be used later for printing graphics hard copies.

The macro file versions numbered 1 or 3 are for use with the longitudinal data, simulated and actual data respectively. The versions numbered 2 or 4 are for use with the lateral-directional data, simulated and actual respectively.

The differences between the versions 1 and 3 and 2 and 4 are relatively small. The biggest differences are that when the actual data (3 and 4) are used, the user is asked for sensor position corrections and queried whether an a-priori weighing vector is to be used during the parameter estimation process by the SSIT. If selected, the weighting input is a vector of the variances for each of the parameters

to be estimated. Therefore, the higher the number, the less the weighting afforded that parameter and vice versa. These M-files are included in Appendix A.

VI. RESULTS AND DISCUSSION

A. SIMULATED DATA

1. Application

The MATLAB M-files LONGDAT.M and LATDIR.M were used to develop the required simulated data representing time history responses. These data included both state and measurement noise. Qualitatively, these data, plotted in Appendix B and discussed below, compare favorably with actual time history data shown in many references. Thus, the model chosen for the simulation was assumed as an adequate mathematical representation of the physical system within the region of applicability, this region being the area or flight regime satisfying the restrictions placed upon the math model during development. Additionally, with the simulation, there is the capability for the user to modify the control inputs and noise magnitudes thereby enhancing the use of these programs as an instructional aid. Various aircraft types with differing inputs and atmospheric conditions potentially could be examined.

The parameter estimation of the simulated data was quite accurate with some exceptions. These exceptions will be discussed later. The accurate parameter estimates, for the given model, validate the MMLE methodology in the presence of

both state and measurement noise. The results show the relative insensitivity of the MMLE method (implemented by the MATLAB SSIT) to noise when the modeling and the excitations (inputs) are chosen with care.

Longitudinal and lateral-directional parameter estimation examples, plots and quantitative output using the simulated data are shown in Appendix B. These simulated examples are for the A-4D and Navion aircraft and the PIONEER UAV. The SSIT quantitative results are tabulated with the initial input parameter estimates and the "truth" or underlying parameters (parameters used to generate the data) for comparison. The tabulated SSIT quantitative results are presented in column format with the column headings as follows: pid, parameter id number; p(pid), final parameter estimate; pref, initial input reference parameter; cramer, Cramer-Rao bounds; 2fcramer, two times a corrected, filtered Cramer-Rao bound; insens, insensitivity, the change in that parameter required to move from the maximum likelihood value to the edge of the confidence ellipsoid. For a single parameter model the insens value is the Cramer-Rao bound.

2. Longitudinal

a. A-4D

The SSIT parameter results for the A-4D, due to the elevator control input shown in Figure B-1, are presented

in Figures B-2 through B-5. These are the input and subsequent response of the A-4D. The α, q, Θ, and acceleration responses correlate well with the truth data used to generate the simulated data. All estimated parameter values for $C_{L\alpha}$, $C_{M\alpha}$, C_{Mq}, $C_{L\delta e}$, and $C_{M\delta e}$ shown in Appendix B are within the 2fcramer bound. This bound represents the 95 percent probability ellipsoid for the parameters. Close inspection of the estimates show that the greatest deviations are for $C_{M\alpha}$ and C_{Mq} and both are within four percent of the actual underlying values used to create the simulation.

b. Navion

The SSIT results for the Navion are shown in the response plots, Figure B-7 through B-10, due to the elevator control doublet input shown in Figure B-6. Again, as was seen in the previous A-4D results, the correlation of the estimated aircraft response plots with the underlying truth derivative responses are good. The largest errors are in the C_{Mq} and $C_{L\delta e}$ derivatives, and they are 12.4 and 7 percent respectively. An accurate prediction of C_{Mq} using theoretical methods is difficult to achieve; Roskam [Ref. 14] notes an acceptable estimated prediction accuracy for this derivative of 20 percent.

c. UAV

The results for the A-4D and Navion are more accurate than the UAV results which are shown in Figures B-11

through B-15. The UAV results demonstrate the importance of designing adequate inputs for each specific vehicle. The input must sufficiently excite the vehicle's dynamic response. Identical elevator input doublets were used for all three vehicles; the UAV elevator input is shown in Figure B-11. The input period is 2 seconds with a maximum amplitude of approximately 4 degrees. The small UAV response due to the small elevator control power ($C_{M\delta_e}$) might be overlooked if the scaling on the response plots were not closely observed. The responses are less than a half to a third those of the A-4D and Navion. These small responses equate to a lower signal-to-noise ratio for the parameter estimator. The response due to the elevator input was not much more significant than the response perceived by the estimator from the state noise (the gust) in combination with the presence of the measurement noise. Thus, the parameters with the exception of $C_{L\alpha}$ (2.4 percent) were all in error greater than 30 percent, and C_{Mq} was 76 percent in error from its underlying truth value. Caution must be exercised in choosing a proper excitation tailored for the particular vehicle's response.

3. Lateral-Directional

The lateral-directional parameter estimates provided by the SSIT for all three simulated aircraft are also shown in Appendix B.

a. A-4D

The A-4D lateral-directional inputs are shown in Figure B-16 and consist of a rudder doublet followed by an aileron doublet. The aircraft responses of β, r, Φ, p, and lateral acceleration due to the aileron and rudder inputs are shown in Figures B-17 through B-21. As can be seen in these plots, the correlation between the measured and estimated responses is good and the parameter estimates which are also shown in Appendix B were accurately estimated. Of the 12 parameters all but three were inside the estimator 2fcramer bound. The three derivatives (i.e. C_{Nr}, $C_{l\delta_a}$ and $C_{N\delta_a}$), although not inside the 2fcramer bound, were very close to the bound, and are within 18, three and eight percent of the truth values respectively. Roskam [Ref. 14 p.236] indicates that the theoretical accuracy for values in estimating C_{Nr} is approximately 25 percent using analytical methods. It is felt that improvements in the estimation of these derivatives could be achieved by investigating different inputs.

b. Navion

The Navion results are also shown in Appendix B, Figures B-23 through B-27 are the response plots due to the rudder and aileron inputs as described by Figure B-22. The inputs are identical to the rudder and aileron inputs used for the A-4D. The Navion aircraft responses β, r, Φ, p, and

lateral acceleration to the inputs (Figures B-23 through B-27) also show good correlation and indicate that the parameter estimates were estimated accurately. A vertical shift in the estimated plot from the true response sometimes occurs and can be misleading. An example of this vertical shift is shown in Figure B-26 by the Navion roll response. This misleading vertical shift is caused by noise on the first data point. The estimated plots were originated by MATLAB at the first data point and any noise in that point causes a vertical shift of the estimated curve. A skilled analyst can adjust the first point to the known initial flight condition and eliminate the misleading vertical offset. The numeric values for the Navion parameter estimates are also shown in Appendix B and were accurately estimated. Of the 12 parameters all were inside the 2fcramer bound.

c. UAV

The parameter estimation results for the UAV due to the rudder and aileron doublets were also quite accurate. The aircraft inputs and response plots are shown in Figures B-28 through B-33. The identical rudder and aileron inputs used for the A-4D and Navion were used for the UAV and are shown in Figure B-28. The misleading vertical shift is more prominent in the UAV responses, especially in Figures B-30 and B-33, the yaw rate and lateral acceleration plots. Again, the responses

correlate well (except for the vertical shift) and the estimated parameters are all within the SSIT 2fcramer bounds.

4. Problems

Significant problems were encountered when trying to employ the SSIT on a very lightly damped or divergent system. This case was experienced with the F104A aircraft. During these conditions the SSIT mmle.m program would halt prior to completion due to an error. The error displayed to the analyst was always a matrix singularity problem encountered during the SSIT computations.

Another rarely encountered problem was the case where the calculated estimates were significantly in error from the actual parameters. This case was very obvious to the user, especially when the estimated plot was compared with the measured data. These significant parameter errors (1 to 2 orders of magnitude) caused the estimated plot to rapidly diverge from the measured data. It is believed that the estimation program was converging upon a local minimum instead of the global minimum of the cost function.

B. ACTUAL FLIGHT DATA

1. Application

The two actual flight tests analyzed were for the longitudinal cases of the F-14A and the T-37 aircraft. The F-14A data were acquired from the Naval Air Test Center and the

T-37 data from NASA-Ames Dryden Flight Research Center. The quantitative results and plots are shown in Appendix B.

a. F-14A

The F-14A elevator input is shown in Figure B-34 with the α, q, Θ and acceleration responses shown in Figures B-35 through B-38. The estimated response period appears to correlate well but the magnitude of the estimated responses are less than the actual measured responses in all cases. This result could be due to the accuracy of the aircraft physical characteristics used in the aircraft model or to data sensor location inaccuracies. These possible problems and the predicted derivatives will be discussed later. The estimation does, however, give a general representation of the aircraft longitudinal characteristics.

b. T-37

The T-37 elevator input is shown in Figure B-39. The aircraft response is shown in Figures B-40 through B-43. The response plots are very accurate with the exception of a disparity in the Θ response (Figure B-42) after approximately the 5 second point. It is not known what caused this perturbation since the q (pitch rate) response correlates well and is the time derivative of Θ response. A possible cause could be attributed to a data acquisition problem, and quite possibly the cause will never be known.

2. Problems

The problems encountered using actual flight data involved obtaining accurate aircraft physical characteristics and arranging the data into the desired format. Since the flight tests were not specifically designed and conducted for this study, much of the required physical data were not readily available and were estimated. The use of estimates was especially true for the F-14A data. Additionally, with no truth data and so many variables on the F-14A such as wing sweep angle, independent left and right horizontal control surfaces and unknown external loadings, the accuracy of the model used and the results are at best an approximation.

In both cases the output parameter estimates were obtained without the use of the program's weighting capability. The parameter estimates appear to be reasonable with the exception of the $C_{L\alpha}$ values, which are inclined towards the higher side of discretion. However, with only one set of data (one flight maneuver) for each airplane analyzed, no significant numerical conclusions can be justifiably deducted. Continued experience with actual flight data will provide a database upon which decisions of proper weighting values can be determined.

VII. CONCLUSIONS AND RECOMMENDATIONS

A. Conclusions

The following conclusions were deduced from this study:

1. The simulated time history data generation and its use in the parameter estimation program worked satisfactorily. The accurate parameter estimates validated the MMLE method as implemented by MATLAB and its relative insensitivity to state and measurement noise if the model and inputs were carefully selected.

2. It is thought that significant benefits can be achieved with the simulated portion of the study as a classroom instructional aid in the Flight Test and Flight Dynamics courses at NPS.

3. The actual flight test data appeared to produce acceptable results. However, many details were not known concerning the data and aircraft characteristics. These unknown details concerned such items as data filtering, sensor position, accurate aircraft physical characteristics, and time lags. Direct involvement with the flight tests, although not essential, would have significant benefits. These benefits would include knowledge of the above missing details, easier data acquisition into the desired data formats and hopefully a more in-depth and accurate analysis.

B. Recommendations

The following recommendations are offered based upon the above conclusions:

1. Incorporate this study into the Flight Test and Flight Dynamics courses at NPS as instructional demonstrations.

2. Continue to develop an on-site data reduction capability to enhance the flight test research being conducted with UAV's at NPS. This development will enable the school to have the capability of comparing computational computer studies and wind-tunnel studies with flight-test results.

3. Investigate the possibility of incorporating the pEst non-linear parameter estimation program (Reference 12) into the flight test research program to cope with any future requirements. These applications could include helicopter dynamics, high-α flight regimes or a host of other areas not particularly well suited to linear modeling.

4. Investigate developing a neural network parameter estimation capability. This capability could be used for development of a real-time reconfigurable flight control system to improve aircraft survivability. These development areas tie parameter estimation into two additional research disciplines of interest at NPS, Neural Networks and Aircraft Combat Survivability.

5. Investigate assimilating the GAT-1 training device and the results from parameter estimation tests into a reconfigurable simulator for instructional purposes in advanced Flight Dynamics, Control and Avionics courses.

6. Investigate the feasibility of using the MATLAB SSIT to accurately determine ship dynamics and perhaps find better ways of controlling unwanted ship motion. The improvement in ship dynamics could lead to improved helicopter landing conditions during rough sea states.

7. Finally, continue the NPS flight test research program on Department of Defense UAV's such as the Pioneer and Exdrone, not only to assist in their evaluation but to stimulate the students' interest with relevant and available military research topics.

LIST OF REFERENCES

1. Mehra, Raman K. and Stepner, David E., "Maximum Likelihood Identification and Optimal Input Design for Identifying Aircraft Stability and Control Derivatives", NASA CR-2200, March 1973.

2. Maine, Richard E. and Iliff, Kenneth W., "Application of Parameter Estimation to Aircraft Stability and Control", NASA RP-1168, June 1986.

3. Lyons, Daniel F., Aerodynamic Analysis of a U.S. Navy and Marine Corps Unmanned Air Vehicle, Master's Thesis, Naval Postgraduate School, Monterey,CA., June 1989.

4. Bray, Robert M., A Wind Tunnel Study of the Pioneer Remotely Piloted Vehicle, Master's Thesis, Naval Postgraduate School, Monterey, CA., June 1991.

5. Redriess, H. A., "An Overview os Parameter Estimation Techniques and Applications in Aircraft Flight Testing", Introductory Paper for Symposium on Parameter Estimation Techniques and Applications in Aircraft Flight Testing, NASA TN D-7647, April 1974.

6. Warner, E. P. and Norton, F. H., "Preliminary Report on Free Flight Tests", NACA Report No. 70, 1919.

7. Soulé, H. A. and Wheatley, J. B., "A Comparison Between the Theoretical and Measured Longitudinal Stability Characteristics of an Airplane", NACA Report No. 442, 1933.

8. Shinbrot, Marvin, "A Least Squares Curve Fitting Method with Applications to Calculation of Stability Coefficients from Transient-Response Data", NACA TN-2341, 1951.

9. Taylor, Lawrence W. Jr. and Iliff, Kenneth W., "A Modified Newton-Raphson Method for Determining Stability Derivatives from Flight Data", Paper presented at Second International Conference on Computing Methods in Optimization Problems (Sanremo, Italy), Sept. 1968.

10. Iliff, Kenneth W. and Maine, Richard E., "Practical Aspects of Using A Maximum Likelihood Estimation Method to Extract Stability and Control Derivatives from Flight Data", NASA TN D-8209, April 1976.

11. Maine, Richard E. and Iliff, Kenneth W., "User's Manual For MMLE3, A General Fortran Program For Maximum Likelihood Parameter Estimation", NASA TP-1563, November 1980.

12. Murray, James E. and Maine, Richard E., "pEst Version 2.1 User's Manual", NASA TM-88280, September 1987.

13. Maine, Richard E. and Iliff, Kenneth W., "Formulation and Implementation of a Practical Algorithm for Parameter Estimation with Process and Measurement Noise", SIAM J. Appl. Math. Vol. 41 No. 3, December 1981.

14. Roskam, Jan, *Aircraft Flight Dynamics and Automatic Flight Controls*, University of Kansas, 1982.

15. Schmidt, Louis V., AE-3340/41 Classroom Notes, Naval Postgraduate School, Monterey, Ca. 1991. (unpublished)

16. Nelson, Robert C., *Flight Stability and Automatic Control, McGraw-Hill*, Inc., 1989.

17. McRuer, Ashkenas, and Graham, *Aircraft Dynamics and Automatic Control*,Princeton University Press, 1973.

18. Hoblit, F.M., *Gust Loads on Aircraft: Concepts and Applications*, AIAA Educational Series 1988.

19. Iliff, Kenneth W. and Maine, Richard E., "More Than You May Want To Know About Maximum Likelihood Estimation", AIAA Paper 84-2070, 1984.

20. Jategaonkar, Ravindra and Plaetschke, Ermin, "Maximum Likelihood Estimation of Parameters in Linear Systems with Process and Measurement Noise", DFVLR 1987.

21. Erickson, Mark S., "Development of a Personnel Computer Based Approach to Aircraft Parameter Identification", Master's Thesis, AFIT, Wright-Patterson AFB, Ohio, 1991.

22. Milne, Garth, *STATE-SPACE IDENTIFICATION TOOL (for use with MATLAB)*, The Math Works, Inc., Sherborn, Ma., 1988.

23. Morelli, Eugene A. and Klein, Vladislav, "Optimal Input Design for Aircraft Parameter Estimation Using Dynamic Programming Principles", AIAA Paper No. 90-2801 CP, August 1990.

24. *MATLAB User's Guide for 80-386-based MS DOS Personal Computers*, The Math Works, Inc., Natick, Ma., 1991.

APPENDIX A -- M-FILES

A. LONGDAT.M

```
clear;
%  MACRO FILE NAME   >======= LONGDAT.M =======<
%                     Date:            1 Feb 92
'-------------------------------------------------------';
disp(ans)
disp(' ')
disp('      MACRO TO GENERATE SIMULATED LONG. DATA       ')
disp('     USING ELEVATOR DOUBLET WITH OR W/O NOISE     ')
disp(ans)
disp(' ')
%------------------ GET AIRCRAFT TYPE TO BE USED ------------
disp('              AIRCRAFT TYPES AVAILABLE             ')
disp(' ')
disp('NAVION   A4D   F104A    JETSTAR    B747    UAV  OTHER   ')
disp(' ')
disp('    SELECT OTHER TO INPUT DATA FOR USER DEFINED AIRCRAFT  ')
disp(' ')
disp('NOTE ========> PROGRAM IS CASE SENSITIVE <========')
disp(' ')
typac=input('TYPE IN DESIRED A/C FROM THE ABOVE LIST.   ','s');
% -------------------------  DETERMINE IF NOISE IS WANTED
disp(' ')
sysn=input('INPUT A 1 TO INCLUDE NOISE AND A ZERO FOR NO NOISE. ');
ndp=201;%       NUMBER OF DATA POINTS -10 SEC
dt=.05;%        TIME STEP FOR THE DATA
amp=.07;%       AMPLITUDE OF DOUBLET (RADIANS)
period=1;%      PERIOD OF DOUBLET IN SEC= PERIOD + 1 SEC
t=[0:ndp-1]*dt;                % TIME VECTOR
simdata=zeros(ndp,5);          % SETUP DATA MATRIX ALL ZEROS
%--------------------------- GENERATE THE INPUT DOUBLET
dslope=(4*amp*dt)/period;
d1=(0:-1*dslope:-1*amp);d1a=-amp*ones(1:10);
d2=(-1*amp+dslope:dslope:amp);d2a=-1*d1a;
d3=(amp-dslope:-1*dslope:0);
simdata(:,1)=[d1 d1a d2 d2a d3 zeros(1,ndp-(period/dt)-21)]';
%----------------- GENERATE THE STABILITY AND CONTROL MATRICIES
disp(' ')
vtrue=input('INPUT AIRCRAFT TRUE VELOCITY      ft/sec   ');
altft=input('INPUT AIRCRAFT ALTITUDE IN        ft       ');
oat=input('INPUT THE OUTSIDE AIR TEMPERATURE   Deg F    ');
if (strcmp('OTHER',typac)>0);
   typac=input('INPUT THE A/C TYPE ....< 6 characters   ','s');
   sref=input('INPUT THE AIRCRAFT REFERENCE AREA   ft^2     ');
   gw=input('INPUT THE AIRCRAFT GROSS WEIGHT       lbs      ');
```

```
   iyy=input('AIRCRAFT Iyy MOMENT OF INERTIA   slug-ft^2    ');
   cbar=input('AIRCRAFT MEAN CHORD LENGTH          ft       ');
   CLa=input('INPUT DERIVATIVE CL_a     1/RAD      ');
   CMa=input('INPUT DERIVATIVE CM_a     1/RAD      ');
   CMq=input('INPUT DERIVATIVE CM_q     1/RAD      ');
   CLde=input('INPUT DERIVATIVE CL_de   1/RAD      ');
   CMde=input('INPUT DERIVATIVE CM_de   1/RAD      ');
   ans=['save  ',typac,'.mat;'];
   eval(ans);  % SAVE THE NEW A/C DATA IN A .MAT FILE
   truth=[CLa CMa CMq CLde CMde];
else
   ans=['load ',typac,'.mat;'];
   eval(ans);
   truth=[CLa CMa CMq CLde CMde];
end
%   CALCULATE DENSITY, DYNAMIC PRESSURE, AND CONSTANTS
rho=.0023769*exp((-1*32.17*altft)/(1716*(oat+460)));
qbar=.5*rho*(vtrue^2);
const1=-1*(qbar*sref)/(gw*vtrue/32.17);
const2=qbar*sref*cbar/iyy;
const3=const2*cbar/(2*vtrue);
const5=qbar*sref/gw;
lw=1750;              % SCALE OF TURBULENCE
   if altft<1750
      lw=altft;
   end
sigw=08;              % RMS VALUE OF TURBULENCE IN FT/SEC
lamda=(vtrue/lw);k=(3*sigw^2)*vtrue/(pi*lw);beta=vtrue/(sqrt(3)*lw)
;
%
a=[const1*CLa  1            0          const1*CLa          0;
   const2*CMa  const3*CMq   0          const2*CMa          0;
   0           1            0          0                   0;
   0           0            0          0                   1;
   0           0            0          -(lamda^2)   -2*lamda];
%
b=[const1*CLde   0                                    0;
   const2*CMde   0                                    0;
   0             0                                    0;
   0             sqrt(k)/vtrue                        0;
   0             (sqrt(k)/vtrue)*(beta-2*lamda)     0];
%
c=[1             0        0            0          0;
   0             1        0            0          0;
   0             0        1            0          0;
   const5*CLa    0        0            0         0];
%
d=[0             0        0;
   0             0        0;
   0             0        0;
```

```
   const5*CLde   0       1];

%
%SIMDATA(:,2)=AOA (RAD)    SIMDATA(:,3)=PITCH RATE (RAD/SEC)
%SIMDATA(:,4)=THETA (RAD) SIMDATA(:,5)=NORMAL ACC (G)
%    RAND NUMBER GENERATOR   MEAN=0     VARIANCE=1
   rand('normal');
[phi,gam]=c2d(a,b,dt);u=[simdata(:,1),sysn*rand(ndp,1),ones(ndp,1)]
;
[ynoise,xnoise]=dlsim(phi,gam,c,d,u);
%              STATENOISE    +   MEASUREMENT NOISE
rand('uniform');
simdata(:,2:3:4:5)=ynoise(:,1:2:3:4) ...
+sysn*[.005818*(rand(ndp,1)-.5) .02909*(rand(ndp,1)-.5)
.005818*(rand(ndp,1)-.5) .01*(rand(ndp,1)-.5)];
%  PLOTS FOR VIEWING ON MONITOR AND STORED IN META FILE
%  ELVATOR vs TIME
plot(t,(180/pi)*simdata(:,1));
xlabel('Time (seconds)');ylabel('Elevator Input (degrees)');
ans=['title(''',typac,'  SIMULATED INPUT'');'];
eval(ans);pause
%meta A:\plots\deltae
% AOA vs TIME
plot(t,(180/pi)*simdata(:,2));
xlabel('Time (seconds)');ylabel('AOA Output (degrees)');
ans=['title(''',typac,'  SIMULATED DATA'');'];
eval(ans);pause
%meta A:\plots\AOA
% PITCH RATE (Q) vs TIME
plot(t,(180/pi)*simdata(:,3));
xlabel('Time (seconds)');ylabel('Pitch Rate, Q, Output (deg/sec)');

ans=['title(''',typac,'  SIMULATED DATA'');'];
eval(ans);pause;
%meta A:\plots\Q
% THETA vs TIME
plot(t,(180/pi)*simdata(:,4));
xlabel('Time (seconds)');ylabel('Theta Output (deg)');
ans=['title(''',typac,'  SIMULATED DATA'');'];
eval(ans);pause;
%meta A:\plots\theta
% NORMAL ACC vs TIME
plot(t,simdata(:,5));
xlabel('Time (seconds)');ylabel('Normal Acceration Output (G)');
ans=['title(''',typac,'  SIMULATED DATA'');'];
eval(ans);pause;
%meta A:\plots\G
disp(' ')
disp('-------------------------------------------------');
disp(' ')
disp('NOTE ====>  DATA BEING SAVED TO A  .mat FILE')
```

```
disp('                    FOR MMLE PROCESSING ')
if sysn>0
 ans=['save N',typac,' typac simdata sysn truth iyy gw sref cbar'];
 eval(ans)
 disp('File name is N followed by the type aircraft.mat N'),typac;
else
 ans=['save NN',typac,' typac simdata sysn truth iyy gw sref
cbar'];
 eval(ans)
 disp('File name is NN followed by the type aircraft.mat
NN'),typac;
end
disp(' ')
!dir/w
disp(' ')
disp(' ')
disp('     NOW RUN npsmmle1.m WITH THE CREATED DATA FILE.')
disp('-------------------------------------------------------');
%--------------------------------------END LONGDAT.M
```

B. LATDIR.M

```
clear;
%  MACRO FILE NAME   >======= LATDIR.M =======<
%                     Date:            3 Feb 92
'-----------------------------------------------';
disp(ans)
disp(' ')
disp(' MACRO TO GENERATE SIMULATED LATERAL-DIRECTIONAL DATA')
disp('   USING AILERON & RUDDER INPUTS WITH OR W/O NOISE')
disp(ans)
disp(' ')
%----------- GET AIRCRAFT TYPE TO BE USED ------------
disp('              AIRCRAFT TYPES AVAILABLE             ')
disp(' ')
disp('NAVION   A4D   F104A    JETSTAR    B747    UAV   OTHER')
disp(' ')
disp('NOTE ========> PROGRAM IS CASE SENSITIVE <========')
disp(' ')
disp('SELECT  "OTHER"  TO INPUT DATA FOR USER DEFINED AIRCRAFT')
disp(' ')
typac=input('TYPE IN DESIRED A/C FROM THE ABOVE LIST.   ','s');
% ------------------------  DETERMINE IF NOISE IS WANTED
disp(' ')
sysn=input('INPUT A 1 TO INCLUDE NOISE AND A ZERO FOR NO NOISE. ');
ndp=301;%------ NUMBER OF DATA POINTS - 15 SEC
dt=.05;%------- TIME STEP FOR THE DATA
amp=.05;%------ AMPLITUDE OF INPUT (RADIANS)
period=1;%----- PERIOD OF DOUBLET IN SEC = PERIOD + 1
t=[0:ndp-1]*dt;%------------- TIME VECTOR
simdata=zeros(ndp,7);%------- SETUP DATA MATRIX ALL ZEROS
%---------------------------- GENERATE THE INPUT DOUBLET
dslope=(4*amp*dt)/period;
d1=(0:-1*dslope:-1*amp);d1a=-amp*ones(1:10);
d2=(-1*amp+dslope:dslope:0);
d3=(dslope:dslope:amp);d3a=-1*d1a;
d4=(amp-dslope:-1*dslope:0);
%----------------------- AILERON AND RUDDER INPUTS
simdata(:,1)=[zeros(1,60) d1 d1a d2 d3 d3a d4
zeros(1,ndp-(period/dt)-81)]';
simdata(:,2)=[-d1 -d1a -d2 -d3 -d3a -d4
zeros(1,ndp-(period/dt)-21)]';
vtrue=input('INPUT AIRCRAFT TRUE VELOCITY    ft/sec      ');
altft=input('INPUT AIRCRAFT ALTITUDE         ft          ');
oat=input('INPUT THE OUTSIDE AIR TEMPERATURE Deg F       ');
if (strcmp('OTHER',typac)>0);
   typac=input('INPUT THE A/C TYPE ....< 6 characters   ','s');
   sref=input('INPUT THE AIRCRAFT REFERENCE AREA  ft^2      ');
   gw=input('INPUT THE AIRCRAFT GROSS WEIGHT      lbs       ');
   ixx=input('AIRCRAFT Ixx MOMENT OF INERTIA   slug-ft^2 ');
```

```
   ixz=input('AIRCRAFT Ixz MOMENT OF INERTIA   slug-ft^2 ');
   izz=input('AIRCRAFT Izz MOMENT OF INERTIA   slug-ft^2 ');
   bbar=input('AIRCRAFT WING SPAN LENGTH "b"   ft        ');
%
   CYb=input('INPUT DERIVATIVE CY_b     1/RAD      ');
   Clb=input('INPUT DERIVATIVE Cl_b     1/RAD      ');
   CNb=input('INPUT DERIVATIVE CN_b     1/RAD      ');
   Clp=input('INPUT DERIVATIVE Cl_p     1/RAD      ');
   CNp=input('INPUT DERIVATIVE CN_p     1/RAD      ');
   Clr=input('INPUT DERIVATIVE Cl_r     1/RAD      ');
   CNr=input('INPUT DERIVATIVE CN_r     1/RAD      ');
   Clda=input('INPUT CONTROL DERIVATIVE Cl_da     1/RAD      ');
   CNda=input('INPUT CONTROL DERIVATIVE CN_da     1/RAD      ');
   CYdr=input('INPUT CONTROL DERIVATIVE CY_dr     1/RAD      ');
   Cldr=input('INPUT CONTROL DERIVATIVE Cl_dr     1/RAD      ');
   CNdr=input('INPUT CONTROL DERIVATIVE CN_dr     1/RAD      ');
   %
   ans=['save ',typac,'.mat;'];
   eval(ans);%---------- SAVE THE NEW A/C DATA IN A .MAT FILE
   truth=[CYb Clb CNb Clp CNp Clr CNr Clda CNda CYdr Cldr CNdr];
else
   ans=['load ',typac,'.mat;'];
   eval(ans);%---------- LOADS THE 'TRUTH DATA' ON SELECTD A/C
   truth=[CYb Clb CNb Clp CNp Clr CNr Clda CNda CYdr Cldr CNdr];
end
%-------------- CALCULATE DENSITY, DYNAMIC PRESSURE, AND MASS
rho=.0023769*exp((-1*32.17*altft)/(1716*(oat+460)));
qbar=.5*rho*vtrue*vtrue;
mass=gw/32.17;
%-------------- CALCULATE CONSTANTS
const1=(qbar*sref)/mass;
const2=qbar*sref*bbar;const2a=const2*bbar/(2*vtrue);
const3=const1/32.17;
%-------------- DRYDEN TURBULENT MODEL CONSTANTS
   if altft<1750
   lw=altft;
   else
      lw=1750;%-------- SCALE OF TURBULENCE

   end
   sigw=05;%---------- RMS VALUE OF TURBULENCE IN FT/SEC

lamda=(vtrue/lw);k=(3*sigw^2)*vtrue/(pi*lw);beta=vtrue/(sqrt(3)*lw)
;
%   SYSTEM STATE SPACE MATRICES
%   INERTIAL MATRIX
In=[1   0   0  0  0  0;
    0 ixx -ixz 0  0  0;
    0 -ixz izz 0  0  0;
    0   0   0  1  0  0;
    0   0   0  0  1  0;
```

```
    0   0   0  0  0  1];
an=[(const1*CYb/vtrue)  0  -1  (32.17/vtrue)  (const1*CYb/vtrue)
0;
    (const2*Clb)  (const2a*Clp)  (const2a*Clr)  0  (const2*Clb)
0;
    (const2*CNb)  (const2a*CNp)  (const2a*CNr)  0  (const2*CNb)
0;
         0              1              0        0        0
0;
         0              0              0        0        0        1

         0              0              0        0   -(lamda^2)
-2*lamda];
%
bn=[0  (const1*CYdr/vtrue)          0              0;
    (const2*Clda) (const2*Cldr)     0              0;
    (const2*CNda) (const2*CNdr)     0              0;
    0         0                     0              0;
    0         0       (sqrt(k)/vtrue)              0;
    0         0   (sqrt(k)/vtrue)*(beta-2*lamda)   0];
%
a=inv(In)*an;b=inv(In)*bn;
c=[1           0    0    0    0        0;
   0           1    0    0    0        0;
   0           0    1    0    0        0;
   0           0    0    1    0        0;
   const3*CYb  0    0    0    0        0];
%
d=[0              0            0             0;
   0              0            0             0;
   0              0            0             0;
   0              0            0             0;
   0              const3*CYdr  0            0];

%
% SIMDATA(:,1)=AILERON INPUT (RAD)  SIMDATA(:,2)=RUDDER INPUT (RAD)
% SIMDATA(:,3)=BETA (RAD)    SIMDATA(:,4)=ROLL RATE (RAD/SEC)
% SIMDATA(:,5)=YAW RATE (RAD/SEC)   SIMDATA(:,6)=ROLL ANGLE (RAD)
% SIMDATA(:,7)=LATERAL ACC
[phi,gam]=c2d(a,b,dt); % ----------   CONVERT TO DISCRETE
%----------- RAND NUMBER GENERATOR   MEAN=0     VARIANCE=1
   rand('normal');
u=[simdata(:,1),simdata(:,2),sysn*rand(ndp,1),ones(ndp,1)];
%--^ INPUTS
[OUTY OUTX]=dlsim(phi,gam,c,d,u);
simdata(:,3:7)=OUTY(:,1:5);
rand('uniform');
%------------- ADD  THE  MEASUREMENT  NOISE
simdata(:,3:7)=simdata(:,3:7)+ ....
```

```
sysn*[.005818*(rand(ndp,1)-.5) .02909*(rand(ndp,1)-.5)
.02909*(rand(ndp,1)-.5) .005818*(rand(ndp,1)-.5)
.01*(rand(ndp,1)-.5)];
%      PLOTS FOR VIEWING ON MONITOR AND STORED IN META FILE
%   AILERON INPUT
subplot(211);plot(t,(180/pi)*simdata(:,1));
xlabel('Time (seconds)');ylabel('Aileron Input (degrees)');
ans=['title(''',typac,'  SIMULATED INPUT'');'];eval(ans);
%  RUDDER INPUT
subplot(212);plot(t,(180/pi)*simdata(:,2));
xlabel('Time (seconds)');ylabel('Rudder Input (degrees)');
ans=['title(''',typac,'  SIMULATED DATA'');'];eval(ans);
pause; %meta A:\plots\Latinput
%  SIDE SLIP (Beta)
subplot(111);plot(t,(180/pi)*simdata(:,3));
xlabel('Time (seconds)');ylabel('Side Slip  ,B, Output (deg)');
ans=['title(''',typac,'  SIMULATED DATA'');'];eval(ans);
pause; %meta A:\plots\sslipout
%  ROLL RATE (P)
plot(t,(180/pi)*simdata(:,4));
xlabel('Time (seconds)');ylabel('Roll Rate  ,P, Output (deg/sec)');
ans=['title(''',typac,'  SIMULATED DATA'');'];eval(ans);
pause; %meta A:\plots\rollrout
% YAW RATE (R)
plot(t,(180/pi)*simdata(:,5));
xlabel('Time (seconds)');ylabel('Yaw Rate  ,R, Output (deg/sec)');
ans=['title(''',typac,'  SIMULATED DATA'');'];eval(ans);
pause; %meta A:\plots\yawout
% ROLL ANGLE (phi)
plot(t,(180/pi)*simdata(:,6));
xlabel('Time (seconds)');ylabel('Roll Angle  Output (deg)');
ans=['title(''',typac,'  SIMULATED DATA'');'];eval(ans);
pause; %meta A:\plots\rollaout
% LATERAL ACCERATION (G)
plot(t,simdata(:,7));
xlabel('Time (seconds)');ylabel('Lateral Acceleration (G)');
ans=['title(''',typac,'  SIMULATED DATA'');'];eval(ans);
pause; %meta A:\plots\gout
clc;
disp('-------------------------------------------------');
disp(' ')
disp('NOTE ====>  DATA BEING SAVED TO A  .mat FILE')
disp('                FOR MMLE PROCESSING ')
if sysn>0
 ans=['save N',typac,' typac simdata sysn truth ixx izz ixz gw sref
bbar'];
 eval(ans)
else
 ans=['save NN',typac,' typac simdata sysn truth ixx izz ixz gw
sref bbar'];
 eval(ans)
```

```
end
disp(' ')
!dir/w
disp('  NOW RUN npsmmle2.m WITH THE CREATED DATA FILE.')
disp('-------------------------------------------------');
%-----------------------------------------END LATDIR.M
```

C. SIMULATED LONGITUDINAL MMLE

1. NPSMMLE1.M

```
%  MACRO NAME        >====== NPSMMLE1.M =======<
%                     Date:              3 Feb 92
clear;
!erase npsmmle1.log;
diary npsmmle1.log
disp('-------------------------------------------------');
disp(' NPS PARAMETER IDENTIFICATION MACRO FILE ')
disp('FOR SIMULATED FLIGHT DATA USING SIMPLFIED SHORT PERIOD ')
disp('      LONGITUDINAL STABILITY AND CONTROL DERIVATIVES')
disp(' ')
disp('-------------------------------------------------');
npsinit1 %------------------  RUN INITIALIZATION MACRO
format compact,clc
load npsinit1;
global sref cbar gw iyy vtrue qbar dt al1 q1 th1 an1 de1;
%------------------------------ INPUT FLIGHT TEST DATA
uydata=zeros(ndp,6);% --------  ESTABLISH DATA MATRIX FOR MMLE.M
%------------------------------ UYDATA(:,1) = DELTA E (INPUT)
%------------------------------ UYDATA(:,3) = AOA
%------------------------------ UYDATA(:,4) = PITCH RATE (Q)
%------------------------------ UYDATA(:,5) = THETA
%------------------------------- UYDATA(:,6) = NORMAL ACC
%------  COLUMN NUMBER ONE IN DATA FILE ELEVATOR INPUT
col1=['uydata(:,1)=simdata(:,1);'];eval(col1);
%------  COLUMN NUMBER TWO IN DATA FILE UNITY INPUT
uydata(:,2)=ones(ndp,1);
%------  COLUMN NUMBER THREE IN DATA FILE ANGLE OF ATTACK
col3=['uydata(:,3)=simdata(:,2);'];eval(col3);
%------  COLUMN NUMBER FOUR IN DATA FILE PITCH RATE
col4=['uydata(:,4)=simdata(:,3);'];eval(col4);
%------  COLUMN NUMBER FIVE IN DATA FILE THETA
col5=['uydata(:,5)=simdata(:,4);'];eval(col5);
%------  COLUMN NUMBER SIX IN DATA FILE NORMAL ACC
col6=['uydata(:,6)=simdata(:,5);'];eval(col6);
%------- INITIAL CONDITIONS
de1=uydata(1,1);al1=uydata(1,3);q1=uydata(1,4);
th1=uydata(1,5);an1=uydata(1,6);
%------- ADDITIONAL INPUTS TO MMLE FOLLOW
p2snam='npsp2ss1'; % -------- MACRO NAME FOR P2SS FUNCTION
p0=pref; %- INITIAL PARAMETER ESTIMATES INPUT DURING NPSINIT1.M
%-- CHECK IF THE WEIGHTING FUNCTION IS TO BE USED FOR INITIAL
VALUES
disp('DO YOU WANT TO WEIGHT THE INITIAL ESTIMATES? INPUT 1=Y 0=NO
')
input(' ');
if ans==1
```

```
  disp('Input Weighting Row Vector length 1 x 5 ')
  disp('Use brackets- ex. [.1 1 1 .1 1]  & lower # higher weight')
  rms0=input(' ');
end
pidq=[1];%--- IDENTIFY WHICH PARAMETERS ARE TO BE IDENTIFIED
pidm=[1:5];%--- IN THE QUADRATIC, MARQUARDT, AND FINAL STAGES.
pidf=[1:5];%- pidq, pidm, pidf MUST BE VALID EVEN IF NOT USED
opt=[0 5 10 10 .02 .005 .001 1];
if sysn==0,opt(4)=0;end
%- DEFAULT ITERATIONS AND CONVERGENCE CRITERIA, IF NOISE FREE
OPT4=0
gg0=eye(4)*(.001);%------------- INNOVATIONS COVARIANCE MATRIX
pert=1e-4;%-PERTURBATION USED FOR NUMERICAL GRADIENT CALCULATION
linesearch=1;%--- USE LINESEARCH TO HELP PROC. NOISE CONVERGENCE
!cls
mmle %----------------- CALL MAIN MMLE MACRO FROM TOOL BOX
%---------------------- PERFORM FINAL CALCULATIONS
cla=pfin(1);cma=pfin(2);cmq=pfin(3);clde=pfin(4);cmde=pfin(5);
disp(' ')
deriv=[cla cma cmq clde cmde];
disp('         MMLE   STABILITY & CONTROL DERIVATIVES       ')
disp('     CLA        CMA        CMQ        CLDE       CMDE')
disp(deriv)
disp('               INITIAL INPUT DERIVATIVES     ')
disp(pref)
if exist('truth')==1;
   disp('  TRUTH DERIVATIVES USED TO GENERATE THE DATA ');
   disp(truth)
end
pause
mleplot1
diary off
%!print npsmmle1.log;
%------------------------------------------ END NPSMMLE1.M
```

2. NPSINIT1.M

```
clear;
% MACRO FILE NAME >======= NPSINIT1.M =======<
%                   Date:              3 Feb 92
% INITIAL SETUP MACRO FOR RUNNING THE PARAMETER
% ESTIMATION TOOLBOX IN MATLAB FOR SIMPLIFIED LONGITUDINAL
% SHORT PERIOD STABILITY AND CONTROL DERIVATIVES
% THIS MACRO 'GETS' CONSTANTS AND DATA FILE
%----- LOAD DATA FILE
disp(' DATA FILE MUST CONTAIN DATA IN COLUMN MATRIX  ')
disp(' MATRIX NAME SIMDATA: N DATA PTS X 5 COLUMNS ')
disp(' ELEVATOR/ALPHA/THETA/PITCH RATE/NORMAL ACC.')
disp(' ')
disp(' DATA FILE NAME--MUST EXIST WITH A  .mat  EXTENSION. ');
data=input('ENTER DATA FILE NAME ( ==> WITH OUT <== .MAT
EXTENSION)?  ','s');
if exist('dt')==0,dt=input('DELTA T BETWEEN DATA POINTS? ');end
ldc=['load ',data,'.mat;'];
eval(ldc);%--EXECUTES LOAD COMMAND
ndp=length(simdata);
t=[0:ndp-1]*dt;
%------- INPUT REQUIRED CONSTANTS--- IF NOT IN DATA FILE
if exist('sref')==0,sref=input('REFERENCE AREA (S) IN SQUARE FEET?
');end
if exist('cbar')==0,cbar=input('MEAN AERODYNAMIC CHORD IN FEET?
');end
if exist('gw')==0,gw=input('AIRCRAFT GROSS WEIGHT IN POUNDS? ');end
if exist('iyy')==0,iyy=input('MOMENT OF INERTIA (IYY) IN SLUG-FT^2?
');end
%sdc=['save  ',data,'  simdata dt sref cbar gw iyy '];eval(sdc);
vtrue=input('AIRSPEED IN FEET PER SECOND? ');
altft=input('AIRCRAFT ALTITUDE IN FEET?  ');
oat=input('OUTSIDE AIR TEMPERATURE?    ');
%------CALCULATE CONSTANTS DENSITY AND DYNAMIC PRESSURE
rho=.0023769*exp((-1*32.174*altft)/(1716*(oat+460)));
qbar=.5*rho*vtrue*vtrue;
%--------------- INPUT INITIAL ESTIMATES FOR STABILITY
%--------------- AND CONTROL DERIVATIVES
pref(1)=input('CL_ALPHA ESTIMATE (1/RAD)? ');
pref(2)=input('CM_ALPHA ESTIMATE (1/RAD)? ');
pref(3)=input('CM_Q ESTIMATE (1/RAD)? ');
pref(4)=input('CL_DE ESTIMATE (1/RAD)? ');
pref(5)=input('CM_DE ESTIMATE (1/RAD)? ');
!erase npsinit1.mat;
save npsinit1
%---------------------------------------- END NPSINIT1.M
```

3. NPSP2SS1.M

```
function [a,phi,gam,c,d,q,x0,dt,rowinq,b]=npsp2ss1(p)
%  MACRO FILE NAME    >======= NPSP2SS1.M =======<
%                        Date:            3 Feb 92
%-------------------------------------------------------
%         MACRO TO EST. FUNCTION FOR TRANSFORMING
%       MODEL PARAMETERS INTO STATE SPACE EQUATIONS
%-------------------------------------------------------
% P2SS FUNCTION FOR NPSMMLE1.M
%-------------------------------------------------------
% p(1) = CL_ALPHA           |  STABILITY AND CONTROL
% p(2) = CM_ALPHA           |      PARAMETERS
% p(3) = CM_Q               |
% p(4) = CL_DE              |
% p(5) = CM_DE              |
%------------------------ PERFORM INITIAL CALCULATIONS
const1=(-1*qbar*sref)/(cos(al1)*gw*vtrue/32.17);
const2=qbar*sref*cbar/iyy;
const3=const2*cbar/(2*vtrue);
const4=32.17*cos(th1)/(vtrue*cos(al1));
const5=(qbar*sref)/gw;
const6=-1*(const1*p(1)*al1+q1+const1*p(4)*de1+const4);
const7=-1*(const2*p(2)*al1+const3*p(3)*q1+const2*p(5)*de1);
const8=(-1*(const5*p(1)*al1+const5*p(4)*de1))+an1;
%------------------------------------ STABILITY DERIVATIVES
a=[const1*p(1)              1            0;
   const2*p(2)          const3*p(3)      0;
   0                        1            0];
%------------------------------------ CONTROL DERIVATIVES
b=[const1*p(4)      (const4+const6);
   const2*p(5)               const7;
   0                       (-1*q1)];
%-------------------------------------MEASUREMENT MATRIX
c=[1                         0            0;
   0                         1            0;
   0                         0            1;
   const5*p(1)               0           0];
%------------------------------------ FEED THROUGH MATRIX
d=[0                     0;
   0                     0;
   0                     0;
   const5*p(4)    const8];
%------------------------------------ STATE NOISE COVARIANCE
q=eye(a)*1e-4;%-- Q IS THE SAME SIZE AS a
%---------------- WITH Q*Q' POS. DEFINITE
%---------------- ROWS IN Q IN WHICH PARAMETERS OCCUR, A VECTOR
%                 SAME DIMENSION AS p
rowinq=[0 0 0 0 0];
```

```
%---------------- INITIAL STATE VECTOR
x0=[al1 q1 th1];
%---------------- DISCRETIZE
% *****NEED TO EDIT dt (below) FOR THE DELTA T OF THE DATA *****
dt=.05;
[phi,gam]=c2d(a,b,dt);
%------------------------------ END NPSP2SS1.M
```

4. MLEPLOT1.M

```
% MACRO FILE NAME        >======= MLEPLOT1.M =======<
%                         Date:              3 Feb 92
% ---- MACRO TO PLOT DATA FROM NPSMMLE1
const1=(-1*qbar*sref)/(cos(al1)*gw*vtrue/32.17);
const2=qbar*sref*cbar/iyy;
const3=const2*cbar/(2*vtrue);
const4=32.17*cos(th1)/(vtrue*cos(al1));
const5=(qbar*sref)/gw;
const6=-1*(const1*pfin(1)*al1+q1+const1*pfin(4)*de1+const4);
const7=-1*(const2*pfin(2)*al1+const3*pfin(3)*q1+const2*pfin(5)*de1)
;
const8=(-1*(const5*pfin(1)*al1+const5*pfin(4)*de1))+an1;
%------------------------------------ STABILITY DERIVATIVES
a=[const1*pfin(1)                    1              0;
   const2*pfin(2)          const3*pfin(3)          0;
   0                              1                0];
%------------------------------------ CONTROL DERIVATIVES
b=[const1*pfin(4)      (const4+const6);
   const2*pfin(5)               const7;
   0                          (-1*q1)];
%-------------------------------------MEASUREMENT MATRIX
c=[1                             0                 0;
   0                             1                 0;
   0                             0                 1;
   const5*pfin(1)                0               0];
%------------------------------------ FEED THROUGH MATRIX
d=[0                       0;
   0                       0;
   0                       0;
   const5*pfin(4)     const8];
rtdc=(180/pi);
%------------- OUT2 = OUTPUT VECTOR---- OUT3 = STATE VECTOR
[OUT2,OUT3]=lsim(a,b,c,d,uydata(:,1:2),t,x0);
if exist('truth')>0;
const6=-1*(const1*truth(1)*al1+q1+const1*truth(4)*de1+const4);
const7=-1*(const2*truth(2)*al1+const3*truth(3)*q1+const2*truth(5)*d
e1);
const8=(-1*(const5*truth(1)*al1+const5*truth(4)*de1))+an1;

a=[const1*truth(1)                   1              0;
   const2*truth(2)          const3*truth(3)          0;
   0                              1                0];
%------------------------------------ CONTROL DERIVATIVES
b=[const1*truth(4)      (const4+const6);
   const2*truth(5)               const7;
   0                          (-1*q1)];
%-------------------------------------MEASUREMENT MATRIX
c=[1                             0                 0;
   0                             1                 0;
```

```
   0                              0              1;
   const5*truth(1)                 0             0];
%------------------------------------ FEED THROUGH MATRIX
d=[0                        0;
   0                        0;
   0                        0;
   const5*truth(4)     const8];
 [TRU2,TRU3]=lsim(a,b,c,d,uydata(:,1:2),t,x0);
end
%---------- PLOTS TO MONITOR AND STORED IN META FILE
if exist('typac')==0,typac=input('INPUT THE AIRCRAFT TYPE ?
',’s’);end
!erase a:\plots\outputg.met;
!erase a:\plots\outputth.met;
!erase a:\plots\outputde.met;
!erase a:\plots\outaoa.met;
!erase a:\plots\outputq.met;
%----ELEVATOR VS TIME
hold off;plot(t,rtdc*uydata(:,1),’-r’);
xlabel('Time (seconds)');ylabel('Elevator Input (degrees)');
ans=['title(’’’,typac,’ ELEVATOR INPUT VS TIME ’’);’];
eval(ans);pause
meta A:\plots\OUTPUTde
%-----AOA (OBSERVED AND ESTIMATED) VS TIME
plot(t,rtdc*uydata(:,3),’*r’);hold on;
xlabel('Time (seconds)');ylabel('AOA (degrees)');
ans=['title(’’’,typac,’  ESTIMATED AND MEASURED AOA RESPONSE’’);’];
eval(ans)
text(.6,.85,’*  Measured Data Points ’,’sc’);pause;
plot(t,rtdc*OUT2(:,1),’og’);
text(.6,.80,’o  Estimated Response ’,’sc’);pause;
if exist('truth')>0;
   plot(t,rtdc*TRU2(:,1),’-b’);
   text(.6,.75,’-  True Response ’,’sc’);pause;
end
pause
meta A:\plots\outAOA
%-----Q (OBSERVED AND ESTIMATED) VS TIME
hold off;plot(t,rtdc*uydata(:,4),’*r’);hold on;
xlabel('Time (seconds)');ylabel('Pitch Rate, Q, (deg/sec)');
ans=['title(’’’,typac,’  ESTIMATED AND MEASURED q RESPONSE’’);’];
eval(ans)
text(.6,.85,’*  Measured Data Points ’,’sc’);pause;
plot(t,rtdc*OUT2(:,2),’og’);
text(.6,.80,’o  Estimated Response ’,’sc’);pause;
if exist('truth')>0;
   plot(t,rtdc*TRU2(:,2),’-b’);
   text(.6,.75,’-  True Response ’,’sc’);pause;
end
pause
meta A:\plots\OUTPUTQ
```

```
%-----THEATA (OBSERVED AND ESTIMATED) VS TIME
hold off;plot(t,rtdc*uydata(:,5),'*r');hold on;
xlabel('Time (seconds)');ylabel('Pitch Angle, Theta, (deg)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED THETA
RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,3),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
if exist('truth')>0;
   plot(t,rtdc*TRU2(:,3),'-b');
   text(.6,.75,'-  True Response ','sc');pause;
end
pause
meta A:\plots\OUTPUTTH
%-----ACCELERATION (OBSERVED AND ESTIMATED) VS TIME
hold off;plot(t,uydata(:,6),'*r');hold on;
xlabel('Time (seconds)');ylabel('Acceleration, G ');
ans=['title(''',typac,'  ESTIMATED AND MEASURED G RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,OUT2(:,4),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
if exist('truth')>0;
   plot(t,TRU2(:,4),'-b');
   text(.6,.75,'-  True Response ','sc');pause;
end
pause
meta A:\plots\OUTPUTG
hold off;
%---------------------------------END MLEPLOT1.M
```

D. SIMULATED LATERAL-DIRECTIONAL MMLE

1. NPSMMLE2.M

```
%  MACRO NAME     >====== NPSMMLE2.M =======<
%                  Date:            3  Feb  92
clear;
!erase npsmmle2.log;
diary npsmmle2.log
'-----------------------------------------------';
disp(ans)
disp('NPS STABILITY PARAMETER IDENTIFICATION MACRO FILE')
disp('              FOR LATERAL-DIRECTIONAL              ')
disp('        STABILITY AND CONTROL DERIVATIVES          ')
disp(' ')
disp(ans)
npsinit2 %-----  RUN INITIALIZATION MACRO
format compact,clc;
load npsinit2;
global sref bbar gw ixx ixz izz vtrue qbar dt beta1 roll1 yaw1
rangle1 ay1 da1 dr1;
%------------------------ INPUT FLIGHT TEST DATA
uydata=zeros(ndp,8);% --- ESTABLISH DATA MATRIX FOR MMLE.M
% UYDATA(:,1) = DELTA Aileron (INPUT)     UYDATA(:,5) = ROLL RATE
(p)
% UYDATA(:,2) = DELTA Rudder (INPUT)      UYDATA(:,6) = YAW RATE (r)
% UYDATA(:,3) = UNITY INPUT               UYDATA(:,7) = ROLL ANGLE
(phi)
% UYDATA(:,4) = BETA (SIDE SLIP)          UYDATA(:,8) = LATERAL G
(ay)
%----  COLUMN NUMBER ONE IN DATA FILE AILERON INPUT
col1=['uydata(:,1)=simdata(:,1);'];eval(col1);
%----  COLUMN NUMBER TWO IN DATA FILE RUDDER INPUT
col2=['uydata(:,2)=simdata(:,2);'];eval(col2);
uydata(:,3)=ones(ndp,1);% UNITY INPUT
%-----  COLUMN NUMBER FOUR IN DATA FILE BETA (SIDE SLIP)
col4=['uydata(:,4)=simdata(:,3);'];eval(col4);
%----  COLUMN NUMBER FIVE IN DATA FILE ROLL RATE (p)
col5=['uydata(:,5)=simdata(:,4);'];eval(col5);
%----  COLUMN NUMBER SIX IN DATA FILE YAW RATE (r)
col6=['uydata(:,6)=simdata(:,5);'];eval(col6);
%----  COLUMN NUMBER SEVEN IN DATA FILE ROLL ANGLE (phi)
col7=['uydata(:,7)=simdata(:,6);'];eval(col7);
%----  COLUMN NUMBER EIGHT IN DATA FILE LATERAL G (ay)
col8=['uydata(:,8)=simdata(:,7);'];eval(col8);
%-----  INITIAL CONDITIONS FOR da, dr, BETA, p, r, phi, ay
beta1=uydata(1,4);roll1=uydata(1,5);yaw1=uydata(1,6);
rangle1=uydata(1,7);ay1=uydata(1,8);
da1=uydata(1,1);dr1=uydata(1,2);
```

```
%------ ADDITIONAL INPUTS TO MMLE FOLLOW
p2snam='npsp2ss2';%----- MACRO NAME FOR P2SS FUNCTION
p0=pref;%---- INITIAL PARAMETER ESTIMATES
%rms0=--- IF USED IT IS THE WEIGHTING FUNCTION FOR INITIAL VALUES
pidq=[1];%--------- IDENTIFY WHICH PARAMETERS ARE TO BE IDENTIFIED
pidm=[1:12];%--------- IN THE QUADRATIC, MARQUARDT, AND FINAL
STAGES.
pidf=[1:12];%------ pidq, pidm, pidf MUST BE VALID EVEN IF NOT USED
opt=[0 5 5 10 .02 .05 .001 1];%- DEFAULT ITERATIONS AND CONVERGENCE
%                                CRITERIA, IF NOISE FREE OPT4=0
if sysn==0,opt(4)=0;end
gg0=eye(5)*(.01);%-------------- INNOVATIONS COVARIANCE MATRIX
pert=1e-4;%-- PERTURBATION USED FOR NUMERICAL GRADIENT CALCULATION
linesearch=1;%----- USE LINESEARCH TO HELP PROC. NOISE CONVERGENCE
mmle%----------------- CALL MAIN MMLE MACRO FROM TOOL BOX
%---------------------- PERFORM FINAL CALCULATIONS
CYb=pfin(1);Clb=pfin(2);CNb=pfin(3);Clp=pfin(4);
CNp=pfin(5);Clr=pfin(6);CNr=pfin(7);Clda=pfin(8);
CNda=pfin(9);CYdr=pfin(10);Cldr=pfin(11);CNdr=pfin(12);
deriv=[CYb       Clb        CNb       Clp       CNp       Clr;
       CNr       Clda       CNda      CYdr      Cldr      CNdr];
disp('         MMLE   STABILITY & CONTROL DERIVATIVES      ')
disp('CY_b       Cl_b       CN_b       Cl_p       CN_p     Cl_r ')
disp(deriv(1,:))
disp('CN_r       Clda       CNda       CYdr       Cldr      CNdr')
disp(deriv(2,:))
disp('               INITIAL INPUT DERIVATIVES     ')
disp(pref(1:6));
disp(pref(7:12));
if exist('truth')==1;
  disp('       "TRUTH DERIVATIVES" USED TO GENERATE DATA ')
  disp(truth(1:6));
  disp(truth(7:12));
end
pause;mleplot2
diary off
%!print npsmmle2.log;
%---------------------------------------- END NPSMMLE2.M
```

2. NPSINIT2.M

```
clear;
% MACRO FILE NAME >======= NPSINIT2.M =======<
%                  Date:            3  Feb 92
% INITIAL SETUP MACRO FOR RUNNING THE PARAMETER
% ESTIMATION TOOLBOX IN MATLAB FOR LATERAL-DIRECTIONAL
% STABILITY AND CONTROL DERIVATIVES
% THIS MACRO 'GETS' CONSTANTS AND DATA FILE
disp(' ')
disp(' DATA FILE NAME--MUST EXIST WITH A  .mat  EXTENSION.  ');
data=input('ENTER DATA FILE NAME ( ==> WITH OUT <== .MAT
EXTENSION)?  ','s');
ldc=['load ',data,'.mat;'];eval(ldc);
ndp=length(simdata);
if exist('dt')==0,dt=input('INPUT dt BETWEEN DATA POINTS.  ');end
t=[0:ndp-1]*dt;
%------------ INPUT REQUIRED CONSTANTS
if exist('sref')==0,sref=input('REFERENCE AREA (S) IN SQUARE FEET?
');end
if exist('bbar')==0,bbar=input('WINGSPAN IN FEET? ');end
if exist('gw')==0,gw=input('AIRCRAFT GROSS WEIGHT IN POUNDS? ');end
if exist('ixx')==0,ixx=input('MOMENT OF INERTIA (Ixx) IN SLUG-FT^2?
');end
if exist('ixz')==0,ixz=input('MOMENT OF INERTIA (Ixz) IN SLUG-FT^2?
');end
if exist('izz')==0,izz=input('MOMENT OF INERTIA (Izz) IN SLUG-FT^2?
');end
%sdc=['save  ',data,'  simdata dt sref bbar gw ixx ixz izz
'];eval(sdc);
vtrue=input('AIRSPEED IN FEET PER SECOND? ');
altft=input('AIRCRAFT ALTITUDE IN FEET?  ');
oat=input('OUTSIDE AIR TEMPERATURE?    ');
rho=.0023769*exp((-1*32.174*altft)/(1716*(oat+460)));
qbar=.5*rho*vtrue*vtrue;
```

```
%-----------------INPUT INITIAL ESTIMATES FOR STABILITY
%-------- AND CONTROL DERIVATIVES TO START MMLE PROGRAM
pref(1)=-.6;% input('CY_beta FROM WIND TUNNEL (1/RAD)? ');
pref(2)=-.15;% input('Cl_beta FROM WIND TUNNEL (1/RAD)? ');
pref(3)=.20;% input('CN_beta FROM WIND TUNNEL (1/RAD)? ');
pref(4)=-.35;% input('Cl_p    FROM WIND TUNNEL (1/RAD)? ');
pref(5)=-.05;% input('CN_p    FROM WIND TUNNEL (1/RAD)? ');
pref(6)=.15;% input('Cl_r    FROM WIND TUNNEL (1/RAD)? ');
pref(7)=-.2;% input('CN_r    FROM WIND TUNNEL (1/RAD)? ');
pref(8)=.05;% input('Cl_da   FROM WIND TUNNEL (1/RAD)? ');
pref(9)=-.001;% input('CN_da   FROM WIND TUNNEL (1/RAD)? ');
pref(10)=.175;%input('CY_dr   FROM WIND TUNNEL (1/RAD)? ');
pref(11)=.02;%input('Cl_dr   FROM WIND TUNNEL (1/RAD)? ');
pref(12)=-.075;%input('CN_dr   FROM WIND TUNNEL (1/RAD)? ');
!erase npsinit2.mat;
save npsinit2
%------------------------------------------ END NPSINIT2.M
```

3. NPSP2SS2.M

```
function [a,phi,gam,c,d,q,x0,dt,rowinq,b]=npsp2ss2(p)
%  MACRO FILE NAME    >======= NPSP2SS2.M =======<
%                       Date:           3  Feb 92
%-------------------------------------------------------
%      MACRO TO ESTABLISH FUNCTION FOR TRANSFORMING
%      MODEL PARAMETERS INTO STATE SPACE EQUATIONS
%-------------------------------------------------------
% P2SS FUNCTION FOR NPSMMLE2.M
%-------------------------------------------------------
% p(1) = CY_beta  |                          | p(7) =  CN_r
% p(2) = Cl_beta  |STABILITY AND CONTROL     | p(8) =  Cl_da
% p(3) = CN_beta  |                          | p(9) =  CN_da
% p(4) = Cl_p     |     PARAMETERS           | p(10) = CY_dr
% p(5) = CN_p     |                          | p(11) = Cl_dr
% p(6) = Cl_r     |                          | p(12) = Cl_dr
%--------------- PERFORM INITIAL CALCULATIONS
const1=(qbar*sref)/(gw/32.17);
const2=qbar*sref*bbar;const2a=const2*bbar/(2*vtrue);
const3=(qbar*sref)/gw;
%----------------- INERTIAL MATRIX
In=[1  0     0  0;
    0  ixx -ixz 0;
    0 -ixz  izz 0;
    0   0    0  1];
%--------------------------- PLANT
an=[const1*p(1)/vtrue          0          -1    (32.17/vtrue);
    const2*p(2)      const2a*p(4)   const2a*p(6)           0;
    const2*p(3)      const2a*p(5)   const2a*p(7)           0;
    0                       1              0               0];
%
a=inv(In)*an;
%
bn=[0       const1*p(10)/vtrue       0;
    const2*p(8)   const2*p(11)       0;
    const2*p(9)   const2*p(12)       0;
    0                      0         0];
%
b=inv(In)*bn;
%
c=[1            0       0      0;
   0            1       0      0;
   0            0       1      0;
   0            0       0      1;
   const3*p(1)  0       0     0];
%
d=[0          0                beta1;
   0          0                roll1;
   0          0                 yaw1;
```

```
   0          0              rangle1;
   0          const3*p(10)      ay1];
%
%----------------------- STATE NOISE COVARIANCE
q=eye(a)*1e-4;%--------- Q IS THE SAME SIZE AS a
%                        WITH Q*Q' POS. DEFINITE!
%---------------- ROWS IN Q IN WHICH PARAMETERS OCCUR, A VECTOR
%                SAME DIMENSION AS p
rowinq=0*p;
%----------------------- INITIAL STATE VECTOR
x0=[beta1 roll1 yaw1 rangle1];
%----------------------- DISCRETIZE
dt=.05;
[phi,gam]=c2d(a,b,dt);
%------------------------------ END NPSP2SS2.M
```

4. MLEPLOT2.M

```
% MACRO FILE NAME   >======= MLEPLOT2.M =======<
%                    Date:            3  Feb  92
%----------------- MACRO TO PLOT DATA FROM NPSMMLE2
%----------------- GENERATE THE PREDICTED DATA
%----------------- CALCULATE STABILITY AND CONTROL MATRIX
mass=gw/32.17;rtdc=(180/pi);const1=(qbar*sref)/mass;
const2=qbar*sref*bbar;const2a=const2*bbar/(2*vtrue);
const3=qbar*sref/gw;
%
In=[1    0    0    0;
    0   ixx -ixz   0;
    0  -ixz  izz   0;
    0    0    0    1];
an=[const1*pfin(1)/vtrue       0              -1      32.17/vtrue;
    const2*pfin(2)      const2a*pfin(4)  const2a*pfin(6)       0;
    const2*pfin(3)      const2a*pfin(5)  const2a*pfin(7)       0;
    0                          1               0                 0];
a=inv(In)*an;
%
bn=[0                  const1*pfin(10)/vtrue          0;
    const2*pfin(8)          const2*pfin(11)           0;
    const2*pfin(9)          const2*pfin(12)           0;
    0                                      0         0];
b=inv(In)*bn;
%
c=[1                0       0       0;
   0                1       0       0;
   0                0       1       0;
   0                0       0       1;
   const3*pfin(1)   0       0      0];
%
d=[0          0                  beta1;
   0          0                  roll1;
   0          0                   yaw1;
   0          0                 rangle1;
   0          const3*pfin(10)     ay1];

% OUT2 = OUTPUTS ---- OUT3 = STATE VECTOR

[OUT2,OUT3]=lsim(a,b,c,d,uydata(:,1:3),t,x0);
if exist('truth')>0
an=[const1*truth(1)/vtrue     0              -1      32.17/vtrue;
    const2*truth(2)      const2a*truth(4)  const2a*truth(6)
0;
    const2*truth(3)      const2a*truth(5)  const2a*truth(7)
0;
    0                          1               0                 0];
a=inv(In)*an;
```

```
%
bn=[0                  const1*truth(10)/vtrue          0;
    const2*truth(8)          const2*truth(11)           0;
    const2*truth(9)          const2*truth(12)           0;
    0                                    0          0];
b=inv(In)*bn;
%
c=[1              0       0       0;
   0              1       0       0;
   0              0       1       0;
   0              0       0       1;
   const3*truth(1)  0       0     0];
%
d=[0          0                    0;
   0          0                    0;
   0          0                    0;
   0          0                    0;
   0          const3*truth(10)     0];
  %
  %TRU2 = OUTPUT  TRU3 = STATE VECTOR
  [TRU2,TRU3]=lsim(a,b,c,d,uydata(:,1:3),t);
end
  %    PLOTS FOR VIEWING ON MONITOR AND STORE TO META FILE
!erase a:\plots\*.met
subplot(211);plot(t,rtdc*uydata(:,1));
xlabel('Time (seconds)');ylabel('Aileron Input (degrees)');
ans=['title(''',typac,' AILERON INPUT VS TIME '');'];eval(ans)
subplot(212);plot(t,rtdc*uydata(:,2));
xlabel('Time (seconds)');ylabel('Rudder Input (degrees)');
ans=['title(''',typac,' RUDDER INPUT VS TIME '');'];eval(ans)
pause; %meta A:\plots\INPUTDAR
%  Beta vs Time
subplot(111);plot(t,rtdc*uydata(:,4),'*r');hold on;
xlabel('Time (seconds)');ylabel('Beta (degrees)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED Beta
RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,1),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
  if exist('truth')>0;
     plot(t,rtdc*TRU2(:,1),'-b');
     text(.6,.75,'-  "True Response"  ','sc');
  end;pause; %meta A:\plots\outbeta
% Roll rate vs Time
hold off;plot(t,rtdc*uydata(:,5),'*r');hold on;
xlabel('Time (seconds)');ylabel('Roll Rate, p, (deg/sec)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED p RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,2),'og');
```

```
text(.6,.80,'o  Estimated Response ','sc');pause;
   if exist('truth')>0;
      plot(t,rtdc*TRU2(:,2),'-b');
      text(.6,.75,'-  "True Response" ','sc');
   end;pause;%meta A:\plots\OUTP
%  Yaw Rate vs Time
hold off;plot(t,rtdc*uydata(:,6),'*r');hold on;
xlabel('Time (seconds)');ylabel('Yaw Rate, r, (deg/sec)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED r RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,3),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
   if exist('truth')>0;
      plot(t,rtdc*TRU2(:,3),'-b');
      text(.6,.75,'-  "True Response" ','sc');
   end;pause;%meta A:\plots\OUTr
% Bank Angle vs Time
hold off;plot(t,rtdc*uydata(:,7),'*r');hold on;
xlabel('Time (seconds)');ylabel('Bank Angle, phi, (deg)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED phi RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,4),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
   if exist('truth')>0;
      plot(t,rtdc*TRU2(:,4),'-b');
      text(.6,.75,'-  "True Response" ','sc');
   end;pause; %meta A:\plots\OUTphi
% Lateral G vs Time
hold off;plot(t,uydata(:,8),'*r');hold on;
xlabel('Time (seconds)');ylabel('Lateral G, ay, (G)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED Lateral G
RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,OUT2(:,5),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
   if exist('truth')>0;
      plot(t,TRU2(:,5),'-b');
      text(.6,.75,'-  "True Response" ','sc');
   end;pause; %meta A:\plots\OUTlatg
hold off;
%----------------------------------------END MLEPLOT2.M
```

E. ACTUAL LONGITUDINAL MMLE

1. NPSMMLE3.M

```
%  MACRO NAME        >====== NPSMMLE3.M =======<
%                     Date:            31 Jan 91
clear;
!erase npsmmle3.log;
diary npsmmle3.log
disp('-------------------------------------------------------');
disp('      NPS PARAMETER IDENTIFICATION MACRO FILE ')
disp('   FOR ACTUAL FLIGHT TESTS USING SIMPLFIED SHORT PERIOD   ')

disp('      LONGITUDINAL STABILITY AND CONTROL DERIVATIVES')
disp(' ')
disp('-------------------------------------------------------');
npsinit3 %------------------------------  RUN INITIALIZATION
MACRO
format compact,clc
load npsinit3;
global sref cbar gw iyy vtrue qbar dt al1 q1 th1 an1 de1;
%--------------------------------------- INPUT FLIGHT TEST DATA
uydata=zeros(ndp,6);% --------  ESTABLISH DATA MATRIX FOR MMLE.M
%-------------------------------- UYDATA(:,1) = DELTA E (INPUT)
%-------------------------------- UYDATA(:,3) = AOA
%-------------------------------- UYDATA(:,4) = PITCH RATE (Q)
%-------------------------------- UYDATA(:,5) = THETA
%-------------------------------- UYDATA(:,6) = NORMAL ACC
%------  COLUMN NUMBER ONE IN DATA FILE ELEVATOR INPUT
col1=['uydata(:,1)=simdata(:,1);'];eval(col1);
%------  COLUMN NUMBER TWO IN DATA FILE UNITY INPUT
uydata(:,2)=ones(ndp,1);
%------  COLUMN NUMBER THREE IN DATA FILE ANGLE OF ATTACK
col3=['uydata(:,3)=simdata(:,2);'];eval(col3);
%------  COLUMN NUMBER FOUR IN DATA FILE PITCH RATE
col4=['uydata(:,4)=simdata(:,3);'];eval(col4);
%------  COLUMN NUMBER FIVE IN DATA FILE THETA
col5=['uydata(:,5)=simdata(:,4);'];eval(col5);
%------  COLUMN NUMBER SIX IN DATA FILE NORMAL ACC
col6=['uydata(:,6)=simdata(:,5);'];eval(col6);
%----SENSOR PLACEMENT CORRECTIONS FOR AOA AND NORMAL ACC
uydata(:,3)=uydata(:,3)+(Xap*uydata(:,4)/vtrue);
for i=1:ndp;
       qsqr(i)=[uydata(i,4)]^2;
       q2=qsqr';
end
uydata(:,6)=uydata(:,6)-(Zan*q2/32.17);
qdot=[diff(uydata(:,4))*(1/dt);0];
uydata(:,6)=uydata(:,6)-(Xan*qdot/32.17);
```

```
%------- INITIAL CONDITIONS
de1=uydata(1,1);al1=uydata(1,3);q1=uydata(1,4);
th1=uydata(1,5);an1=uydata(1,6);
%------- ADDITIONAL INPUTS TO MMLE FOLLOW
p2snam='npsp2ss3'; % -------- MACRO NAME FOR P2SS FUNCTION
p0=pref; %- INITIAL PARAMETER ESTIMATES INPUT DURING NPSINIT3.M
%-- CHECK IF THE WEIGHTING FUNCTION IS TO BE USED FOR INITIAL
VALUES
disp('DO YOU WANT TO WEIGHT THE INITIAL ESTIMATES? INPUT 1=Y 0=NO
')
input(' ');
if ans==1
  disp('Input Weighting Row Vector length 1 x 5 ')
  disp('Use brackets- ex. [.1 1 1 .1 1]  & lower # higher weight')
  rms0=input(' ');
end
pidq=[1];%--- IDENTIFY WHICH PARAMETERS ARE TO BE IDENTIFIED
pidm=[1:5];%--- IN THE QUADRATIC, MARQUARDT, AND FINAL STAGES.
pidf=[1:5];%- pidq, pidm, pidf MUST BE VALID EVEN IF NOT USED
opt=[0 5 5 10 .02 .10 .001 1];
%- DEFAULT ITERATIONS AND CONVERGENCE CRITERIA, IF NOISE FREE
OPT4=0
gg0=eye(4)*(.001);%------------- INNOVATIONS COVARIANCE MATRIX
pert=1e-4;%-PERTURBATION USED FOR NUMERICAL GRADIENT CALCULATION
linesearch=1;%--- USE LINESEARCH TO HELP PROC. NOISE CONVERGENCE
!cls
mmle %--------------------- CALL MAIN MMLE MACRO FROM TOOL BOX
%------------------------- PERFORM FINAL CALCULATIONS
cla=pfin(1);cma=pfin(2);cmq=pfin(3);clde=pfin(4);cmde=pfin(5);
disp(' ')
disp(' ')
deriv=[cla cma cmq clde cmde];
disp('         MMLE   STABILITY & CONTROL DERIVATIVES       ')
disp('     CLA        CMA        CMQ        CLDE       CMDE')
disp(deriv)
disp('               INITIAL INPUT DERIVATIVES     ')
disp(pref)
pause
mleplot3
diary off
%!print npsmmle3.log;
%------------------------------------------ END NPSMMLE3.M
```

2. NPSINIT3.M

```
clear;
% MACRO FILE NAME >======= NPSINIT3.M =======<
%                   Date:              31 Dec 91
% INITIAL SETUP MACRO FOR RUNNING THE PARAMETER
% ESTIMATION TOOLBOX IN MATLAB FOR SIMPLIFIED LONGITUDINAL
% SHORT PERIOD STABILITY AND CONTROL DERIVATIVES
% THIS MACRO 'GETS' CONSTANTS AND DATA FILE
%----- LOAD DATA FILE
disp(' DATA FILE MUST CONTAIN DATA IN COLUMN MATRIX  ')
disp(' MATRIX NAME SIMDATA: N DATA PTS X 5 COLUMNS ')
disp(' ELEVATOR/ALPHA/THETA/PITCH RATE/NORMAL ACC.')
disp(' ')
disp(' DATA FILE NAME--MUST EXIST WITH A  .mat  EXTENSION. ');
data=input('ENTER DATA FILE NAME ( ==> WITH OUT <== .MAT
EXTENSION)?  ','s');
if exist('dt')==0,dt=input('DELTA T BETWEEN DATA POINTS? ');end
ldc=['load ',data,'.mat;'];
eval(ldc);%--EXECUTES LOAD COMMAND
ndp=length(simdata);
t=[0:ndp-1]*dt;
%------- INPUT REQUIRED CONSTANTS--- IF NOT IN DATA FILE
if exist('sref')==0,sref=input('REFERENCE AREA (S) IN SQUARE FEET?
');end
if exist('cbar')==0,cbar=input('MEAN AERODYNAMIC CHORD IN FEET?
');end
if exist('gw')==0,gw=input('AIRCRAFT GROSS WEIGHT IN POUNDS? ');end
if exist('iyy')==0,iyy=input('MOMENT OF INERTIA (IYY) IN SLUG-FT^2?
');end
if exist('Xap')==0,Xap=input('X-DIST FROM cg TO AOA PROBE (FT
+FWD)');end
if exist('Zan')==0,Zan=input('Z-DIST FROM cg TO NORMAL ACCEL (FT
+DWN)');end
if exist('Xan')==0,Xan=input('X-DIST FROM cg TO NORMAL ACCEL (FT
+FWD)');end
%sdc=['save  ',data,'  simdata dt sref cbar gw iyy Xap Zan
Xan'];eval(sdc);
vtrue=input('AIRSPEED IN FEET PER SECOND? ');
altft=input('AIRCRAFT ALTITUDE IN FEET?  ');
oat=input('OUTSIDE AIR TEMPERATURE?    ');
%------CALCULATE CONSTANTS DENSITY AND DYNAMIC PRESSURE
rho=.0023769*exp((-1*32.174*altft)/(1716*(oat+460)));
qbar=.5*rho*vtrue*vtrue;
```

```
%--------------- INPUT INITIAL ESTIMATES FOR STABILITY
%--------------- AND CONTROL DERIVATIVES
pref(1)=input('CL_ALPHA ESTIMATE (1/RAD)? ');
pref(2)=input('CM_ALPHA ESTIMATE (1/RAD)? ');
pref(3)=input('CM_Q ESTIMATE (1/RAD)? ');
pref(4)=input('CL_DE ESTIMATE (1/RAD)? ');
pref(5)=input('CM_DE ESTIMATE (1/RAD)? ');
!erase npsinit3.mat;
save npsinit3
%------------------------------------------- END NPSINIT3.M
```

3. NPSP2SS3.M

```
function [a,phi,gam,c,d,q,x0,dt,rowinq,b]=npsp2ss3(p)
%  MACRO FILE NAME    >======= NPSP2SS3.M =======<
%                        Date:           31 Jan 92
%-------------------------------------------------------------------

%         MACRO TO EST. FUNCTION FOR TRANSFORMING
%       MODEL PARAMETERS INTO STATE SPACE EQUATIONS
%-------------------------------------------------------------------

% P2SS FUNCTION FOR NPSMMLE3.M
%-------------------------------------------------------------------
% p(1) = CL_ALPHA           |  STABILITY AND CONTROL
% p(2) = CM_ALPHA           |      PARAMETERS
% p(3) = CM_Q               |
% p(4) = CL_DE              |
% p(5) = CM_DE              |
%------------------------------------ PERFORM INITIAL CALCULATIONS
const1=(-1*qbar*sref)/(cos(al1)*gw*vtrue/32.17);
const2=qbar*sref*cbar/iyy;
const3=const2*cbar/(2*vtrue);
const4=32.17*cos(th1)/(vtrue*cos(al1));
const5=(qbar*sref)/gw;
const6=-1*(const1*p(1)*al1+q1+const1*p(4)*de1+const4);
const7=-1*(const2*p(2)*al1+const3*p(3)*q1+const2*p(5)*de1);
const8=(-1*(const5*p(1)*al1+const5*p(4)*de1))+an1;
%------------------------------------ STABILITY DERIVATIVES
a=[const1*p(1)              1            0;
   const2*p(2)          const3*p(3)      0;
   0                        1            0];
%------------------------------------ CONTROL DERIVATIVES
b=[const1*p(4)      (const4+const6);
   const2*p(5)               const7;
   0                       (-1*q1)];
%-------------------------------------MEASUREMENT MATRIX
c=[1                         0            0;
   0                         1            0;
   0                         0            1;
   const5*p(1)               0           0];
%------------------------------------ FEED THROUGH MATRIX
d=[0                     0;
   0                     0;
   0                     0;
   const5*p(4)    const8];
%------------------------------------ STATE NOISE COVARIANCE
q=eye(a)*1e-4;%-- Q IS THE SAME SIZE AS a
%---------------- WITH Q*Q' POS. DEFINITE
%---------------- ROWS IN Q IN WHICH PARAMETERS OCCUR, A VECTOR
%                 SAME DIMENSION AS p
```

```
rowinq=[0 0 0 0 0];
%---------------- INITIAL STATE VECTOR
x0=[a11 q1 th1];
%---------------- DISCRETIZE
% *****NEED TO EDIT dt (below) FOR THE DELTA T OF THE DATA *****
dt=.1;
[phi,gam]=c2d(a,b,dt);
%------------------------------ END NPSP2SS3.M
```

4. MLEPLOT3.M

```
% MACRO FILE NAME        >======= MLEPLOT3.M =======<
%                         Date:              31 Jan 92
% ---- MACRO TO PLOT DATA FROM NPSMMLE3
const1=(-1*qbar*sref)/(cos(al1)*gw*vtrue/32.17);
const2=qbar*sref*cbar/iyy;
const3=const2*cbar/(2*vtrue);
const4=32.17*cos(th1)/(vtrue*cos(al1));
const5=(qbar*sref)/gw;
const6=-1*(const1*pfin(1)*al1+q1+const1*pfin(4)*de1+const4);
const7=-1*(const2*pfin(2)*al1+const3*pfin(3)*q1+const2*pfin(5)*de1)
;
const8=(-1*(const5*pfin(1)*al1+const5*pfin(4)*de1))+an1;
%------------------------------------ STABILITY DERIVATIVES
a=[const1*pfin(1)                  1              0;
   const2*pfin(2)          const3*pfin(3)        0;
   0                              1              0];
%------------------------------------ CONTROL DERIVATIVES
b=[const1*pfin(4)      (const4+const6);
   const2*pfin(5)               const7;
   0                          (-1*q1)];
%-------------------------------------MEASUREMENT MATRIX
c=[1                              0              0;
   0                              1              0;
   0                              0              1;
   const5*pfin(1)                 0             0];
%------------------------------------ FEED THROUGH MATRIX
d=[0                        0;
   0                        0;
   0                        0;
   const5*pfin(4)     const8];
rtdc=(180/pi);
%------------- OUT2 = OUTPUT VECTOR---- OUT3 = STATE VECTOR
[OUT2,OUT3]=lsim(a,b,c,d,uydata(:,1:2),t,x0);
%---------- PLOTS TO MONITOR AND STORED IN META FILE
if exist('typac')==0,typac=input('INPUT THE AIRCRAFT TYPE ?
','s');end
!erase a:\plots\outputg.met;
!erase a:\plots\outputth.met;
!erase a:\plots\outputde.met;
!erase a:\plots\outaoa.met;
!erase a:\plots\outputq.met;
%----ELEVATOR VS TIME
hold off;plot(t,rtdc*uydata(:,1),'-r');
xlabel('Time (seconds)');ylabel('Elevator Input (degrees)');
ans=['title(''',typac,' ELEVATOR INPUT VS TIME '');'];
eval(ans)
pause
meta A:\plots\OUTPUTde
```

```
%-----AOA (OBSERVED AND ESTIMATED) VS TIME
plot(t,rtdc*uydata(:,3),'*r');hold on;
xlabel('Time (seconds)');ylabel('AOA (degrees)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED AOA RESPONSE'');'];

eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,1),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause
meta A:\plots\outAOA
%-----Q (OBSERVED AND ESTIMATED) VS TIME
hold off;plot(t,rtdc*uydata(:,4),'*r');hold on;
xlabel('Time (seconds)');ylabel('Pitch Rate, Q, (deg/sec)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED q RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,2),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause
meta A:\plots\OUTPUTQ
%-----THEATA (OBSERVED AND ESTIMATED) VS TIME
hold off;plot(t,rtdc*uydata(:,5),'*r');hold on;
xlabel('Time (seconds)');ylabel('Pitch Angle, Theta, (deg)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED THETA
RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,3),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause
meta A:\plots\OUTPUTTH
%-----ACCELERATION (OBSERVED AND ESTIMATED) VS TIME
hold off;plot(t,uydata(:,6),'*r');hold on;
xlabel('Time (seconds)');ylabel('Acceleration, G ');
ans=['title(''',typac,'  ESTIMATED AND MEASURED G RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,OUT2(:,4),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause
meta A:\plots\OUTPUTG
hold off;
%-----------------------------------------END MLEPLOT3.M
```

F. ACTUAL LATERAL-DIRECTIONAL MMLE

1. NPSMMLE4.M

```
%  MACRO NAME     >====== NPSMMLE4.M =======<
%                  Date:            5  Feb  92
clear;
!erase npsmmle4.log;
diary npsmmle4.log
'-------------------------------------------------';
disp(ans)
disp('NPS STABILITY PARAMETER IDENTIFICATION MACRO FILE')
disp('              FOR LATERAL-DIRECTIONAL              ')
disp('       STABILITY AND CONTROL DERIVATIVES           ')
disp(' ')
disp(ans)
npsinit4 %-----  RUN INITIALIZATION MACRO
format compact,clc;
load npsinit4;
global sref bbar gw ixx ixz izz vtrue qbar dt beta1 roll1 yaw1
rangle1 ay1 da1 dr1;
%------------------------ INPUT FLIGHT TEST DATA
uydata=zeros(ndp,8);% --- ESTABLISH DATA MATRIX FOR MMLE.M
% UYDATA(:,1) = DELTA Aileron (INPUT)     UYDATA(:,5) = ROLL RATE
(p)
% UYDATA(:,2) = DELTA Rudder (INPUT)      UYDATA(:,6) = YAW RATE (r)
% UYDATA(:,3) = UNITY INPUT               UYDATA(:,7) = ROLL ANGLE
(phi)
% UYDATA(:,4) = BETA (SIDE SLIP)          UYDATA(:,8) = LATERAL G
(ay)
%----  COLUMN NUMBER ONE IN DATA FILE AILERON INPUT
col1=['uydata(:,1)=simdata(:,1);'];eval(col1);
%----  COLUMN NUMBER TWO IN DATA FILE RUDDER INPUT
col2=['uydata(:,2)=simdata(:,2);'];eval(col2);
uydata(:,3)=ones(ndp,1);% UNITY INPUT
%----- COLUMN NUMBER FOUR IN DATA FILE BETA (beta)
col4=['uydata(:,4)=simdata(:,3);'];eval(col4);
%----  COLUMN NUMBER FIVE IN DATA FILE ROLL RATE (p)
col5=['uydata(:,5)=simdata(:,4);'];eval(col5);
%----  COLUMN NUMBER SIX IN DATA FILE YAW RATE (r)
col6=['uydata(:,6)=simdata(:,5);'];eval(col6);
%----  COLUMN NUMBER SEVEN IN DATA FILE ROLL ANGLE (phi)
col7=['uydata(:,7)=simdata(:,6);'];eval(col7);
%----  COLUMN NUMBER EIGHT IN DATA FILE LATERAL G (ay)
col8=['uydata(:,8)=simdata(:,7);'];eval(col8);
%----- SENSOR PLACEMENT CORRECTIONS FOR beta AND ay
uydata(:,4)=uydata(:,4)-(Xbp/vtrue)*uydata(:,6);
   rdot=[diff(uydata(:,6))*(1/dt);0];
   pdot=[diff(uydata(:,5))*(1/dt);0];
```

```
uydata(:,8)=uydata(:,8)-(rdot*Xay/32.17)+(pdot*Zay/32.17);
%----- INITIAL CONDITIONS FOR da, dr, BETA, p, r, phi, ay
beta1=uydata(1,4);roll1=uydata(1,5);yaw1=uydata(1,6);
rangle1=uydata(1,7);ay1=uydata(1,8);
da1=uydata(1,1);dr1=uydata(1,2);
%------ ADDITIONAL INPUTS TO MMLE FOLLOW
p2snam='npsp2ss4';%----- MACRO NAME FOR P2SS FUNCTION
p0=pref;%---- INITIAL PARAMETER ESTIMATES
%--CHECK IF WEIGHTING FUNCTION DESIRED
disp('Do you want to WEIGHT the initial estimates? INPUT 1=YES
0=NO');
  ans=input(' ');
  if ans==1,
     disp('Input WEIGHTING ROW MATRIX:  1 X 12')
     disp('USE BRACKETS- ex. [.1 .1 1 1 1 10 1 .1 1 1 .1 1]');
     disp('NOTE *** The LOWER the # the HIGHER the WEIGHTING!');
     rms0=input(' ');
  end;clc;
pidq=[1];%--------- IDENTIFY WHICH PARAMETERS ARE TO BE IDENTIFIED
pidm=[1:12];%--------- IN THE QUADRATIC, MARQUARDT, AND FINAL
STAGES.
pidf=[1:12];%------ pidq, pidm, pidf MUST BE VALID EVEN IF NOT USED
opt=[0 5 5 10 .02 .10 .001 1];%- DEFAULT ITERATIONS AND CONVERGENCE
%                                CRITERIA, IF NOISE FREE OPT4=0
gg0=eye(5)*(.01);%-------------- INNOVATIONS COVARIANCE MATRIX
pert=1e-4;%-- PERTURBATION USED FOR NUMERICAL GRADIENT CALCULATION
linesearch=1;%----- USE LINESEARCH TO HELP PROC. NOISE CONVERGENCE
mmle%----------------- CALL MAIN MMLE MACRO FROM TOOL BOX
%---------------------- PERFORM FINAL CALCULATIONS
CYb=pfin(1);Clb=pfin(2);CNb=pfin(3);Clp=pfin(4);
CNp=pfin(5);Clr=pfin(6);CNr=pfin(7);Clda=pfin(8);
CNda=pfin(9);CYdr=pfin(10);Cldr=pfin(11);CNdr=pfin(12);
deriv=[CYb       Clb        CNb       Clp       CNp       Clr;
       CNr       Clda       CNda      CYdr      Cldr      CNdr];
disp('        MMLE   STABILITY & CONTROL DERIVATIVES      ')
disp('CY_b       Cl_b       CN_b       Cl_p       CN_p     Cl_r ')
disp(deriv(1,:))
disp('CN_r       Clda       CNda       CYdr       Cldr      CNdr')
disp(deriv(2,:))
disp('               INITIAL INPUT DERIVATIVES     ')
disp(pref(1:6));
disp(pref(7:12));
pause;mleplot4
diary off
%!print npsmmle4.log;
%---------------------------------------- END NPSMMLE4.M
```

2. NPSINIT4.M

```
clear;
% MACRO FILE NAME >======= NPSINIT4.M =======<
%                    Date:              5  Feb 92
% INITIAL SETUP MACRO FOR RUNNING THE PARAMETER
% ESTIMATION TOOLBOX IN MATLAB FOR LATERAL-DIRECTIONAL
% STABILITY AND CONTROL DERIVATIVES
% THIS MACRO 'GETS' CONSTANTS AND DATA FILE
disp(' ')
disp('  DATA FILE NAME--MUST EXIST WITH A  .mat  EXTENSION. ');
disp('    DATA FILE MUST CONTAIN DATA IN COLUMN MATRIX');
disp('   MATRIX NAME SIMDATA:  N (data pts) X 7 COLUMNS');
disp('AILERON/RUDDER/BETA/ROLL RATE/YAW RATE/ROLL ANGLE/LAT G');
data=input('ENTER DATA FILE NAME ( ==> WITH OUT <== .MAT
EXTENSION)?  ','s');
ldc=['load ',data,'.mat;'];eval(ldc);
ndp=length(simdata);
if exist('dt')==0,dt=input('INPUT dt BETWEEN DATA POINTS.  ');end
t=[0:ndp-1]*dt;
%------------- INPUT REQUIRED CONSTANTS
if exist('sref')==0,sref=input('REFERENCE AREA (S) IN SQUARE FEET?
');end
if exist('bbar')==0,bbar=input('WINGSPAN IN FEET? ');end
if exist('gw')==0,gw=input('AIRCRAFT GROSS WEIGHT IN POUNDS? ');end
if exist('ixx')==0,ixx=input('MOMENT OF INERTIA (Ixx) IN SLUG-FT^2?
');end
if exist('ixz')==0,ixz=input('MOMENT OF INERTIA (Ixz) IN SLUG-FT^2?
');end
if exist('izz')==0,izz=input('MOMENT OF INERTIA (Izz) IN SLUG-FT^2?
');end
if exist('Xbp')==0,Xbp=input('X-DIST FROM cg TO BETA PROBE (FT
+FWD)');end
if exist('Zay')==0,Zay=input('Z-DIST FROM cg TO LATERAL ACC. (FT
+DWN)');end
if exist('Xay')==0,Xay=input('X-DIST FROM cg TO LATERAL ACC. (FT
+FWD)');end
%sdc=['save  ',data,'  simdata dt sref bbar gw ixx ixz izz Xbp Zay
Xay'];
%eval(sdc);
vtrue=input('AIRSPEED IN FEET PER SECOND? ');
altft=input('AIRCRAFT ALTITUDE IN FEET?  ');
oat=input('OUTSIDE AIR TEMPERATURE?    ');
rho=.0023769*exp((-1*32.174*altft)/(1716*(oat+460)));
qbar=.5*rho*vtrue*vtrue;
%-----------------INPUT INITIAL ESTIMATES FOR STABILITY
%-------- AND CONTROL DERIVATIVES TO START MMLE PROGRAM
pref(1)=-.6;% input('CY_beta FROM WIND TUNNEL (1/RAD)? ');
pref(2)=-.15;% input('Cl_beta FROM WIND TUNNEL (1/RAD)? ');
pref(3)=.20;% input('CN_beta FROM WIND TUNNEL (1/RAD)? ');
```

```
pref(4)=-.35;% input('Cl_p    FROM WIND TUNNEL (1/RAD)? ');
pref(5)=-.05;% input('CN_p    FROM WIND TUNNEL (1/RAD)? ');
pref(6)=.15;% input('Cl_r    FROM WIND TUNNEL (1/RAD)? ');
pref(7)=-.2;% input('CN_r    FROM WIND TUNNEL (1/RAD)? ');
pref(8)=.05;% input('Cl_da   FROM WIND TUNNEL (1/RAD)? ');
pref(9)=-.001;% input('CN_da   FROM WIND TUNNEL (1/RAD)? ');
pref(10)=.175;%input('CY_dr   FROM WIND TUNNEL (1/RAD)? ');
pref(11)=.02;%input('Cl_dr   FROM WIND TUNNEL (1/RAD)? ');
pref(12)=-.075;%input('CN_dr   FROM WIND TUNNEL (1/RAD)? ');
!erase npsinit4.mat;
save npsinit4
%---------------------------------------- END NPSINIT4.M
```

3. NPSP2SS4.M

```
function [a,phi,gam,c,d,q,x0,dt,rowinq,b]=npsp2ss4(p)
%  MACRO FILE NAME    >======= NPSP2SS4.M =======<
%                        Date:           5  Feb 92
%-------------------------------------------------------
%      MACRO TO ESTABLISH FUNCTION FOR TRANSFORMING
%      MODEL PARAMETERS INTO STATE SPACE EQUATIONS
%-------------------------------------------------------
% P2SS FUNCTION FOR NPSMMLE4.M
%-------------------------------------------------------
% p(1) = CY_beta  |                        | p(7) =  CN_r
% p(2) = Cl_beta  |STABILITY AND CONTROL   | p(8) =  Cl_da
% p(3) = CN_beta  |                        | p(9) =  CN_da
% p(4) = Cl_p     |     PARAMETERS         | p(10) = CY_dr
% p(5) = CN_p     |                        | p(11) = Cl_dr
% p(6) = Cl_r     |                        | p(12) = Cl_dr
%--------------- PERFORM INITIAL CALCULATIONS
const1=(qbar*sref)/(gw/32.17);
const2=qbar*sref*bbar;const2a=const2*bbar/(2*vtrue);
const3=(qbar*sref)/gw;
%----------------- INERTIAL MATRIX
In=[1   0     0   0;
    0   ixx -ixz  0;
    0  -ixz  izz  0;
    0    0     0   1];
%--------------------------- PLANT
an=[const1*p(1)/vtrue          0          -1     (32.17/vtrue);
    const2*p(2)       const2a*p(4)   const2a*p(6)           0;
    const2*p(3)       const2a*p(5)   const2a*p(7)           0;
    0                        1              0                0];
%
a=inv(In)*an;
%
bn=[0        const1*p(10)/vtrue        0;
    const2*p(8)    const2*p(11)        0;
    const2*p(9)    const2*p(12)        0;
    0                         0        0];
%
b=inv(In)*bn;
%
c=[1             0        0       0;
   0             1        0       0;
   0             0        1       0;
   0             0        0       1;
   const3*p(1)   0        0      0];
%
d=[0           0                beta1;
   0           0                roll1;
   0           0                 yaw1;
```

```
   0          0              rangle1;
   0          const3*p(10)      ay1];
%
%----------------------- STATE NOISE COVARIANCE
q=eye(a)*1e-4;%--------- Q IS THE SAME SIZE AS a
%                        WITH Q*Q' POS. DEFINITE!
%---------------- ROWS IN Q IN WHICH PARAMETERS OCCUR, A VECTOR
%                 SAME DIMENSION AS p
rowinq=0*p;
%----------------------- INITIAL STATE VECTOR
x0=[beta1 roll1 yaw1 rangle1];
%----------------------- DISCRETIZE
%*****NOTE***** CHANGE dt TO THE ACTUAL DATA VALUE ****
dt=.05;
[phi,gam]=c2d(a,b,dt);
%------------------------------ END NPSP2SS4.M
```

4. MLEPLOT4.M

```
% MACRO FILE NAME   >======= MLEPLOT4.M =======<
%                    Date:            5  Feb  92
%----------------- MACRO TO PLOT DATA FROM NPSMMLE4
%----------------- GENERATE THE PREDICTED DATA
%----------------- CALCULATE STABILITY AND CONTROL MATRIX
mass=gw/32.17;rtdc=(180/pi);const1=(qbar*sref)/mass;
const2=qbar*sref*bbar;const2a=const2*bbar/(2*vtrue);
const3=qbar*sref/gw;
%
In=[1   0   0   0;
    0  ixx -ixz 0;
    0 -ixz  izz 0;
    0   0   0   1];
an=[const1*pfin(1)/vtrue       0              -1      32.17/vtrue;
    const2*pfin(2)      const2a*pfin(4)  const2a*pfin(6)       0;
    const2*pfin(3)      const2a*pfin(5)  const2a*pfin(7)       0;
    0                          1               0                 0];
a=inv(In)*an;
%
bn=[0                  const1*pfin(10)/vtrue          0;
    const2*pfin(8)          const2*pfin(11)           0;
    const2*pfin(9)          const2*pfin(12)           0;
    0                                    0          0];
b=inv(In)*bn;
%
c=[1                0       0       0;
   0                1       0       0;
   0                0       1       0;
   0                0       0       1;
   const3*pfin(1)   0       0      0];
%
d=[0          0                  beta1;
   0          0                  roll1;
   0          0                   yaw1;
   0          0                rangle1;
   0          const3*pfin(10)     ay1];

% OUT2 = OUTPUTS ---- OUT3 = STATE VECTOR

[OUT2,OUT3]=lsim(a,b,c,d,uydata(:,1:3),t,x0);
  %    PLOTS FOR VIEWING ON MONITOR AND STORE TO META FILE
!erase a:\plots\*.met
subplot(211);plot(t,rtdc*uydata(:,1));
xlabel('Time (seconds)');ylabel('Aileron Input (degrees)');
ans=['title(''',typac,' AILERON INPUT VS TIME '');'];eval(ans)
subplot(212);plot(t,rtdc*uydata(:,2));
xlabel('Time (seconds)');ylabel('Rudder Input (degrees)');
```

```
ans=['title(''',typac,' RUDDER INPUT VS TIME '');'];eval(ans)
pause; %meta A:\plots\INPUTDAR
%  Beta vs Time
subplot(111);plot(t,rtdc*uydata(:,4),'*r');hold on;
xlabel('Time (seconds)');ylabel('Beta (degrees)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED Beta
RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,1),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause; %meta A:\plots\outbeta
% Roll rate vs Time
hold off;plot(t,rtdc*uydata(:,5),'*r');hold on;
xlabel('Time (seconds)');ylabel('Roll Rate, p, (deg/sec)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED p RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,2),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause;%meta A:\plots\OUTP
%  Yaw Rate vs Time
hold off;plot(t,rtdc*uydata(:,6),'*r');hold on;
xlabel('Time (seconds)');ylabel('Yaw Rate, r, (deg/sec)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED r RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,3),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause;%meta A:\plots\OUTr
% Bank Angle vs Time
hold off;plot(t,rtdc*uydata(:,7),'*r');hold on;
xlabel('Time (seconds)');ylabel('Bank Angle, phi, (deg)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED phi RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,rtdc*OUT2(:,4),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause; %meta A:\plots\OUTphi
```

```
% Lateral G vs Time
hold off;plot(t,uydata(:,8),'*r');hold on;
xlabel('Time (seconds)');ylabel('Lateral G, ay, (G)');
ans=['title(''',typac,'  ESTIMATED AND MEASURED Lateral G
RESPONSE'');'];
eval(ans)
text(.6,.85,'*  Measured Data Points ','sc');pause;
plot(t,OUT2(:,5),'og');
text(.6,.80,'o  Estimated Response ','sc');pause;
pause; %meta A:\plots\OUTlatg
hold off;
%--------------------------------------END MLEPLOT4.M
```

APPENDIX B OUTPUT

A. SIMULATED OUTPUT

1. Longitudinal

a. *A-4D*

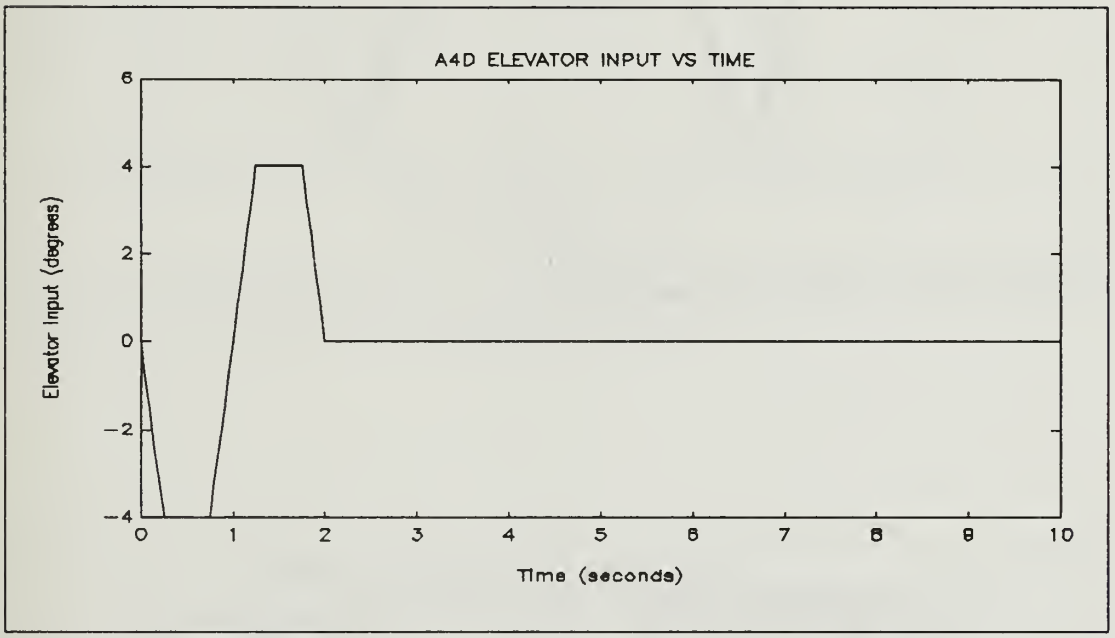

Figure B-1 A-4D Elevator Input

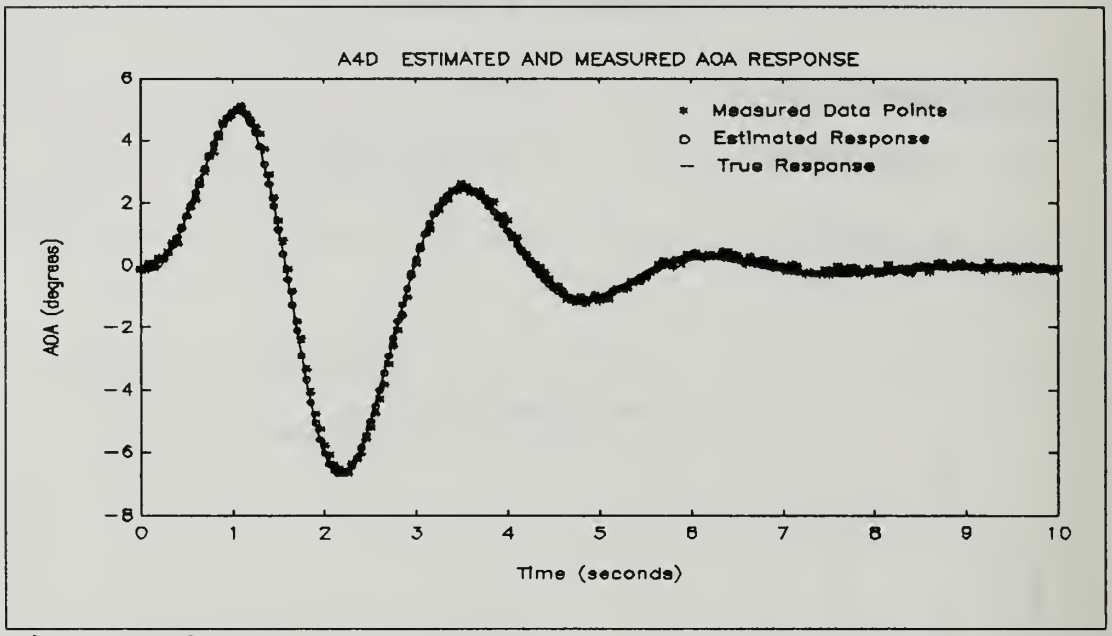

Figure B-2 A-4D AOA Response

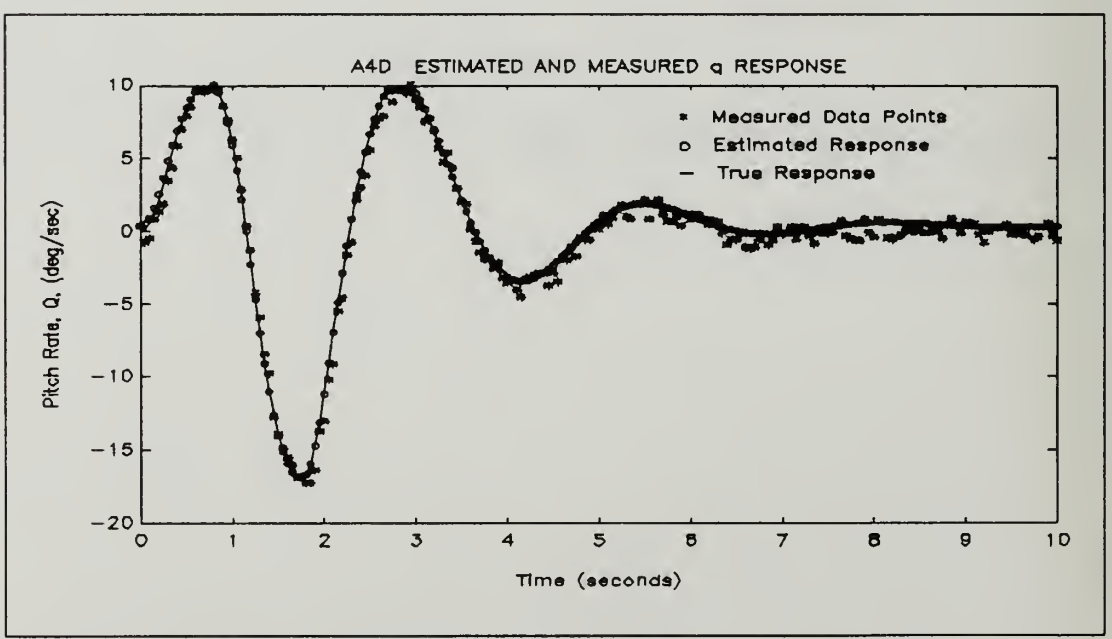

Figure B-3 A-4D Pitch Rate Response

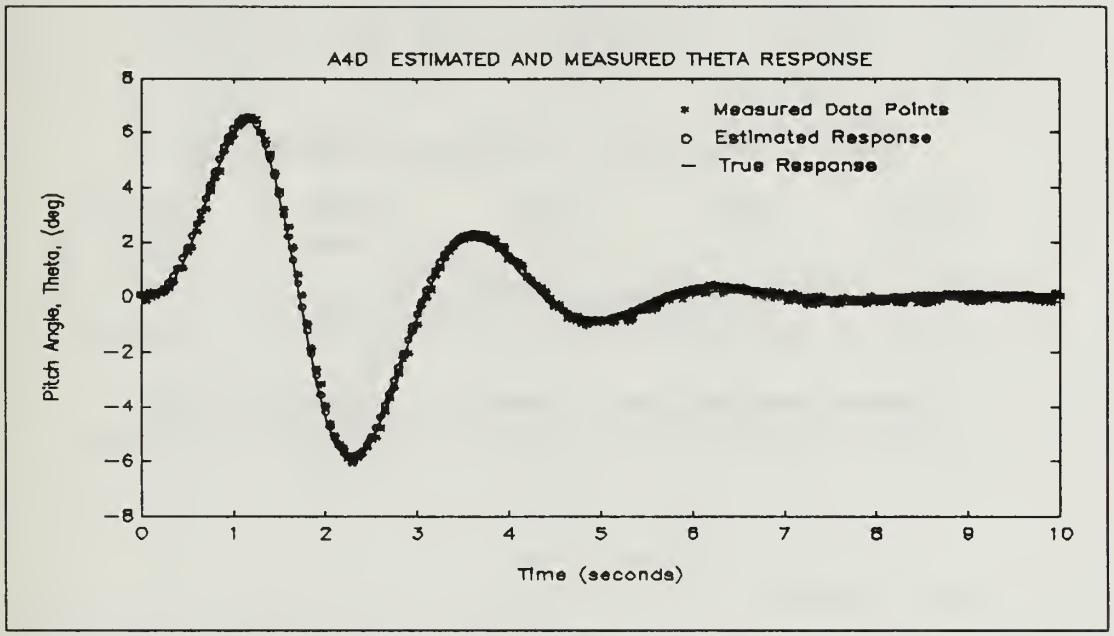

Figure B-4 A-4D Pitch Angle Response

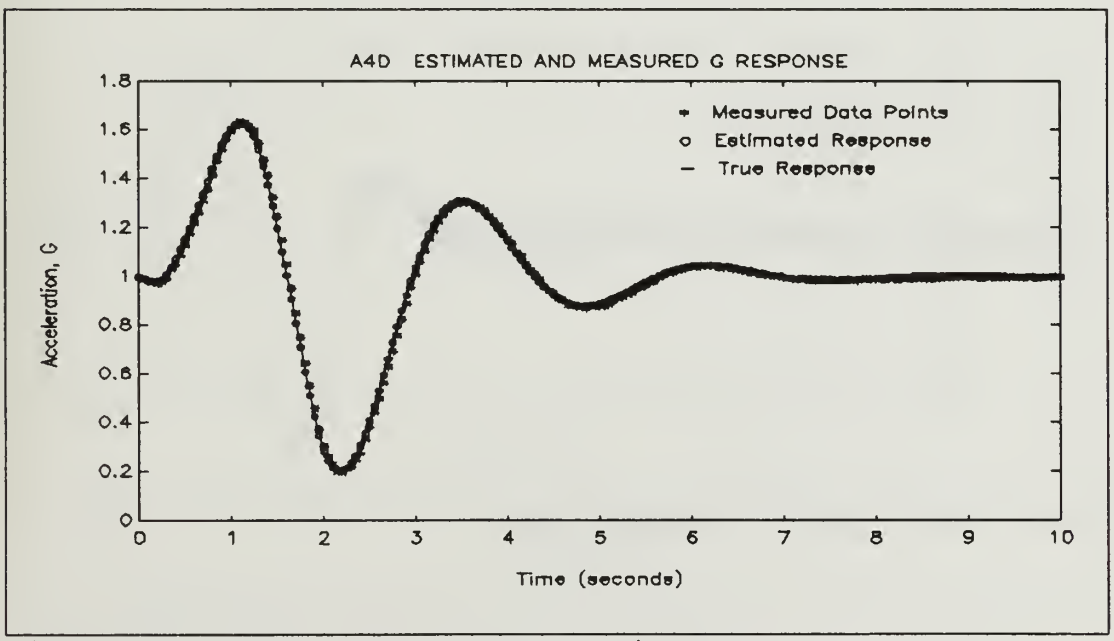

Figure B-5 A-4D Normal Acceleration Response

A-4D SIMULATED LONGITUDINAL RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	3.4568	5.5000	0.0090	0.0231	0.0074
2.0000	-0.3818	-0.8000	0.0007	0.0018	0.0006
3.0000	-3.4717	-15.0000	0.0311	0.0795	0.0187
4.0000	0.3441	0.8000	0.0102	0.0260	0.0099
5.0000	-0.4905	-1.5000	0.0020	0.0052	0.0011

MMLE STABILITY & CONTROL DERIVATIVES

CLA	CMA	CMQ	CLDE	CMDE
3.4568	-0.3818	-3.4717	0.3441	-0.4905

INITIAL INPUT DERIVATIVES

5.5000	-0.8000	-15.0000	0.8000	-1.5000

"TRUTH DERIVATIVES" USED TO GENERATE DATA

3.4500	-0.3800	-3.6000	0.3600	-0.5000

b. NAVION

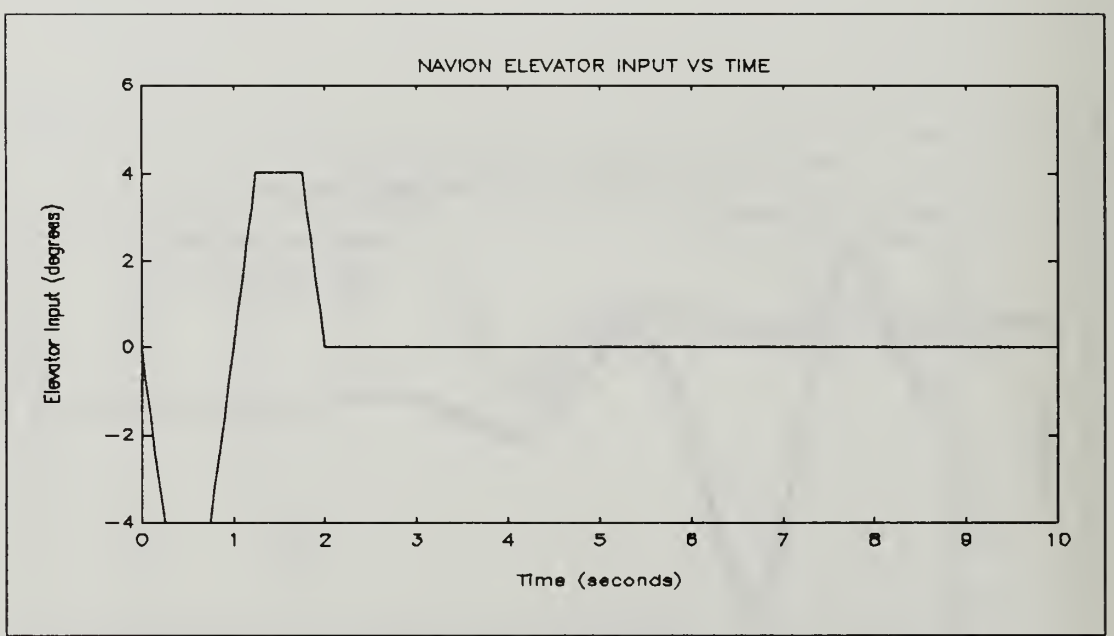

Figure B-6 Navion Elevator Input

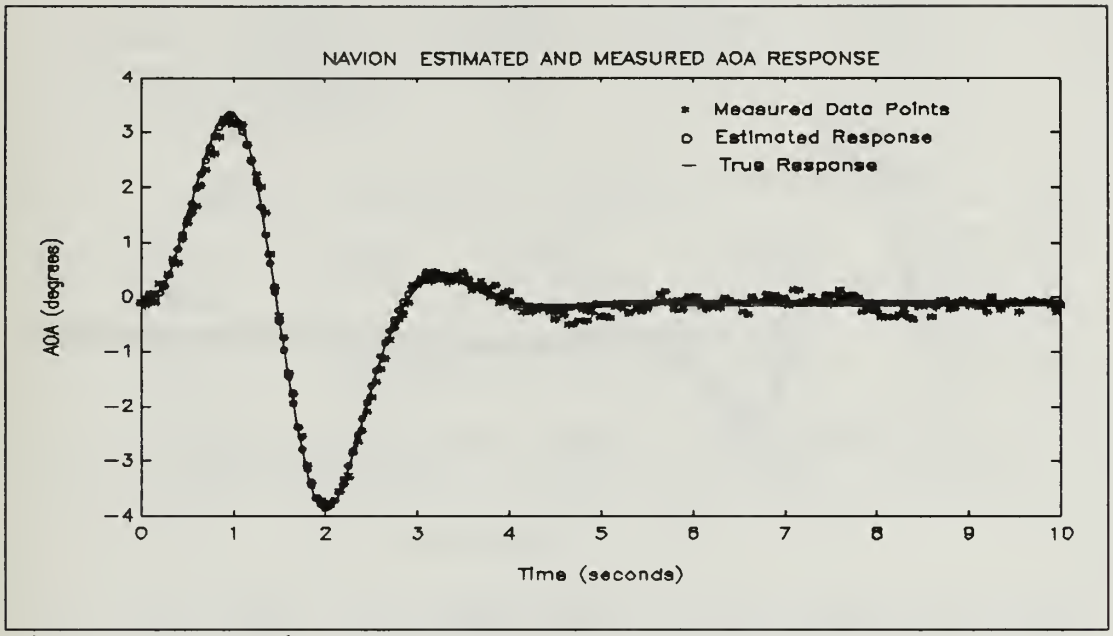

Figure B-7 Navion AOA Response

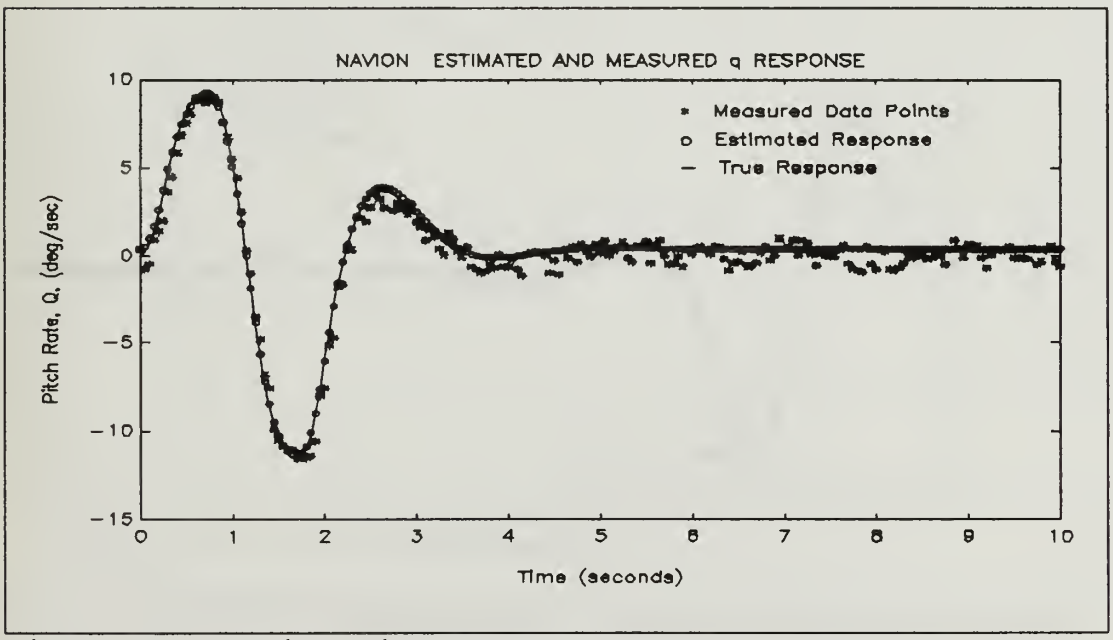

Figure B-8 Navion Pitch Rate Response

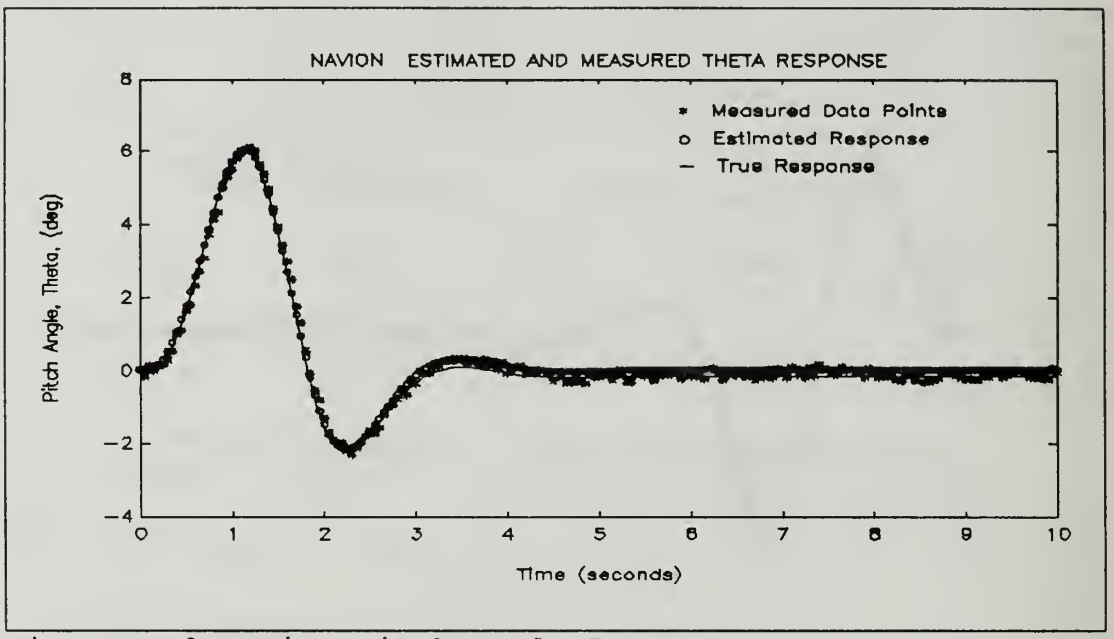

Figure B-9 Navion Pitch Angle Response

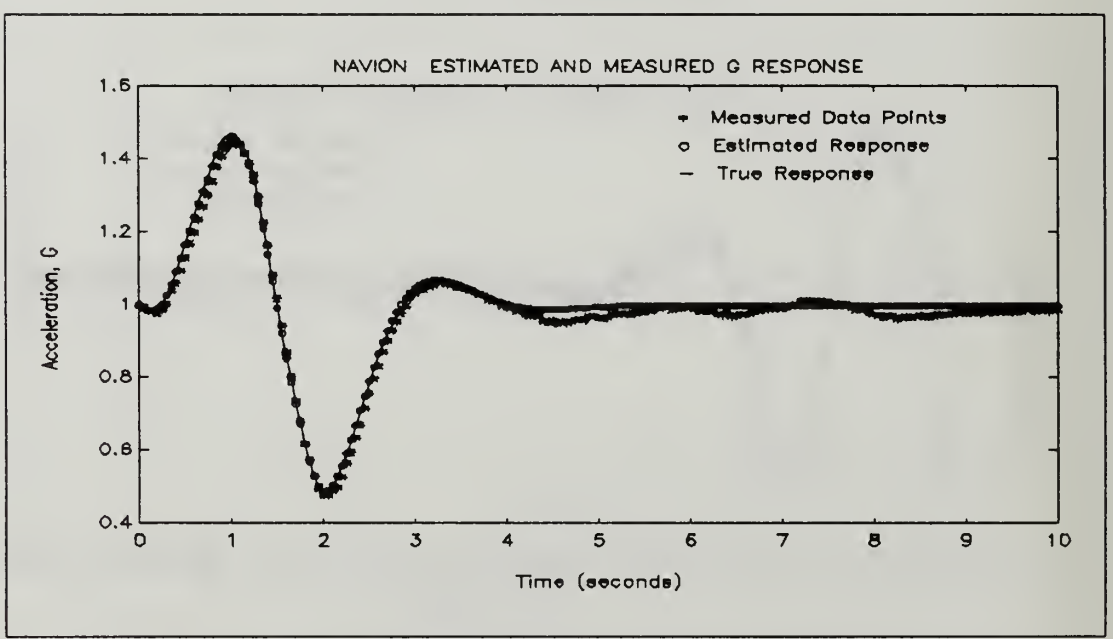

Figure B-10 Navion Normal Acceleration Response

NAVION SIMULATED LONGITUDINAL RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	4.4567	5.0000	0.0251	0.0716	0.0227
2.0000	-0.7097	-0.5000	0.0064	0.0184	0.0046
3.0000	-8.7238	-15.0000	0.2638	0.7538	0.0895
4.0000	0.3300	0.5000	0.0201	0.0573	0.0185
5.0000	-0.8596	-1.0000	0.0107	0.0307	0.0041

MMLE STABILITY & CONTROL DERIVATIVES

CLA	CMA	CMQ	CLDE	CMDE
4.4567	-0.7097	-8.7238	0.3300	-0.8596

INITIAL INPUT DERIVATIVES

5.0000	-0.5000	-15.0000	0.5000	-1.0000

"TRUTH DERIVATIVES" USED TO GENERATE DATA

4.4400	-0.6830	-9.9600	0.3550	-0.9230

c. UAV

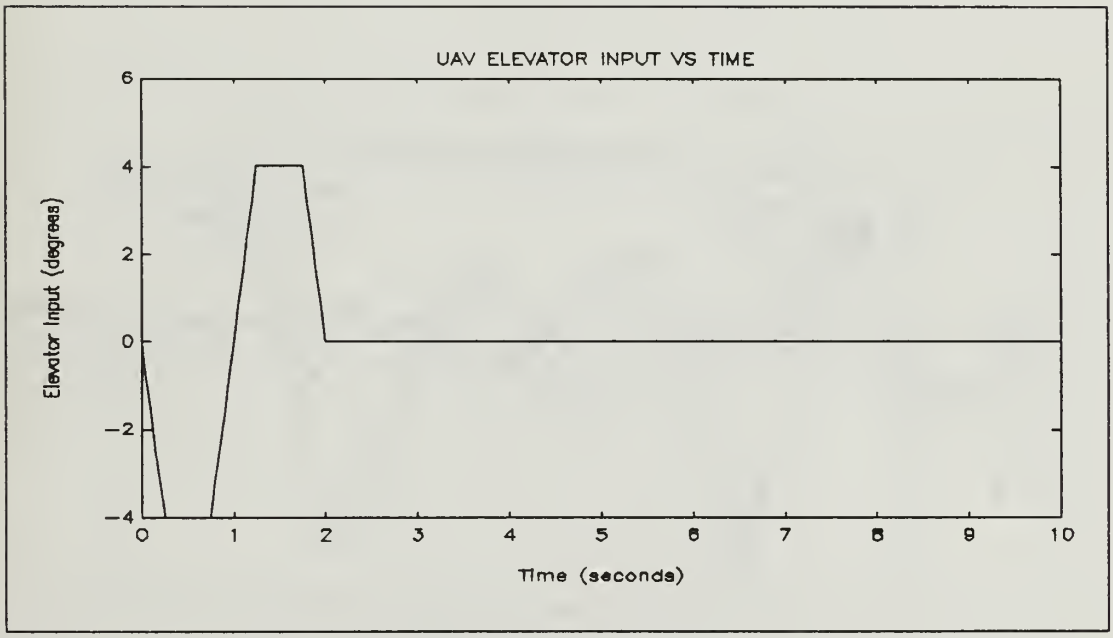

Figure B-11 UAV Elevator Input

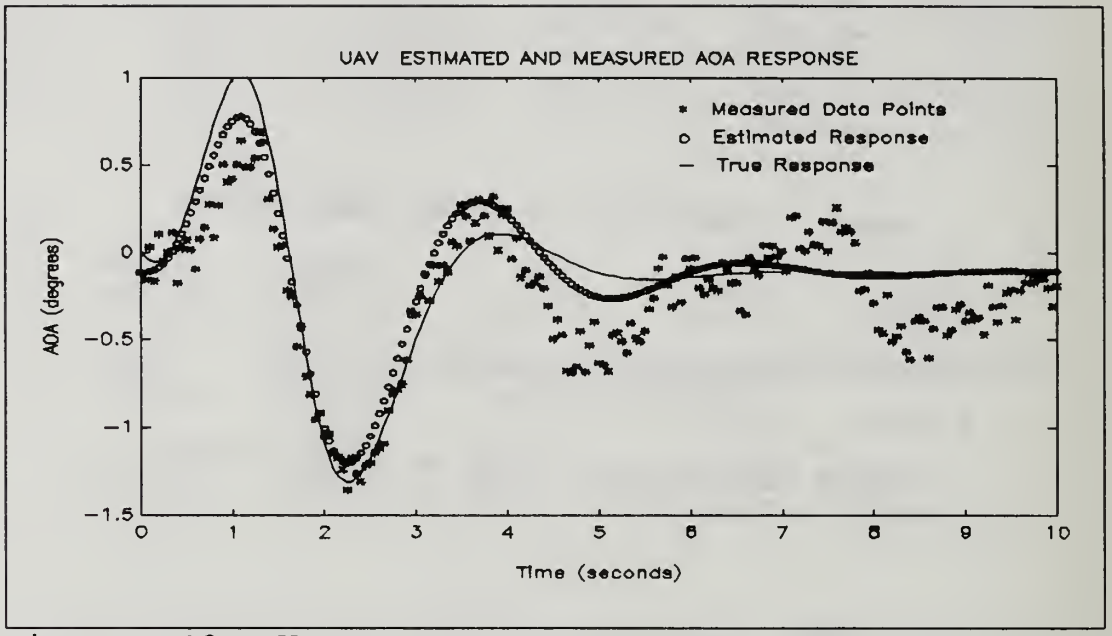

Figure B-12 UAV AOA Response

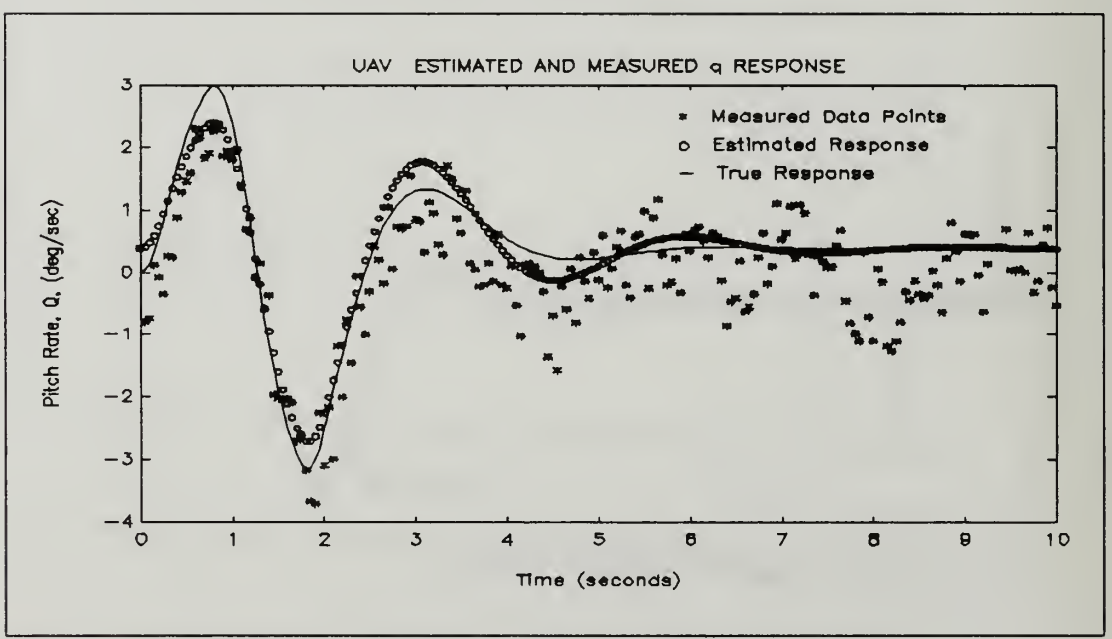

Figure B-13 UAV Pitch Rate Response

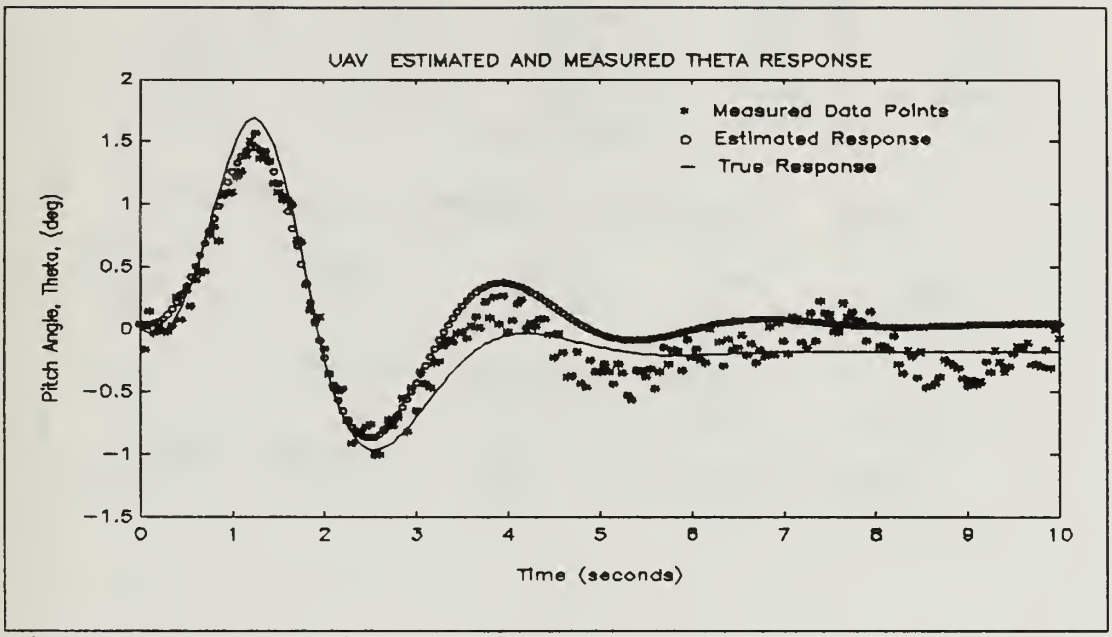

Figure B-14 UAV Pitch Angle Response

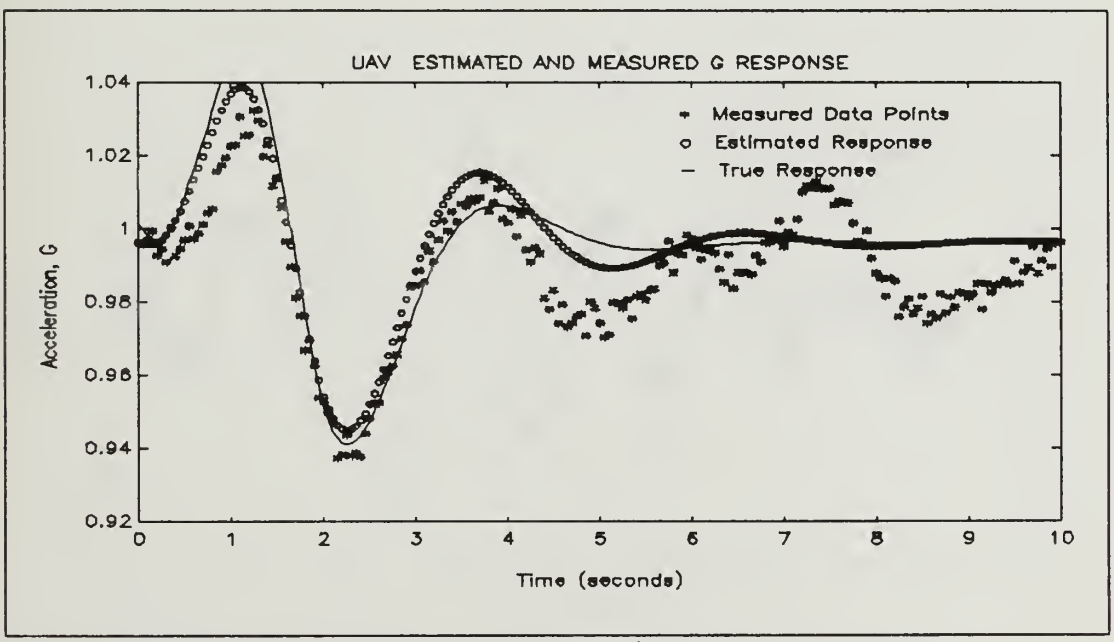

Figure B-15 UAV Normal Acceleration Response

UAV SIMULATED LONGITUDINAL RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	5.1314	6.0000	0.0989	0.2913	0.0888
2.0000	-0.9390	-0.6000	0.0244	0.0718	0.0177
3.0000	-3.4725	-10.0000	1.2628	3.7180	0.6143
4.0000	0.0473	0.5000	0.0248	0.0731	0.0232
5.0000	-0.2188	-1.1000	0.0096	0.0283	0.0055

MMLE STABILITY & CONTROL DERIVATIVES

CLA	CMA	CMQ	CLDE	CMDE
5.1314	-0.9390	-3.4725	0.0473	-0.2188

INITIAL INPUT DERIVATIVES

6.0000	-0.6000	-10.0000	0.5000	-1.1000

"TRUTH DERIVATIVES" USED TO GENERATE DATA

5.0100	-0.6900	-14.7500	0.0760	-0.3333

2. Lateral-Directional

a. A-4D

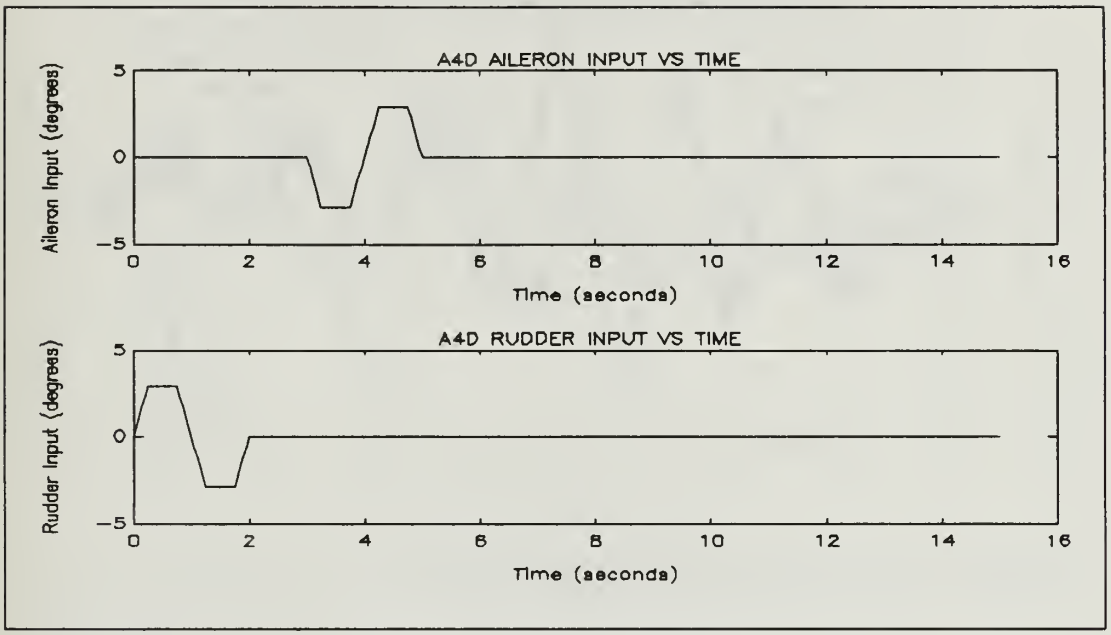

Figure B-16 A-4D Aileron and Rudder Inputs

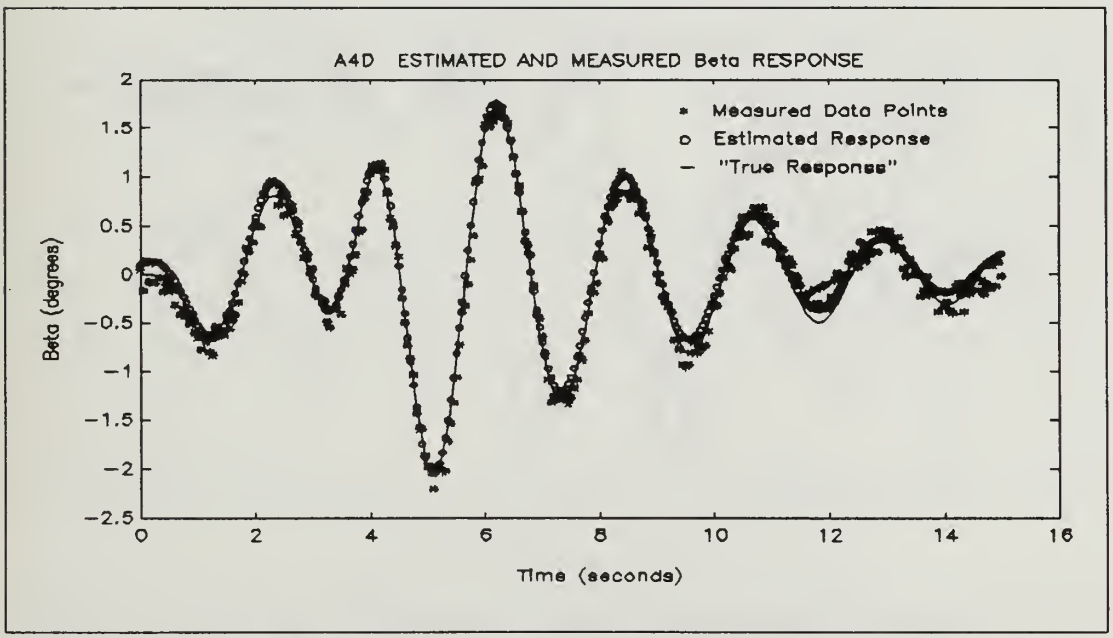

Figure B-17 A-4D Beta Response

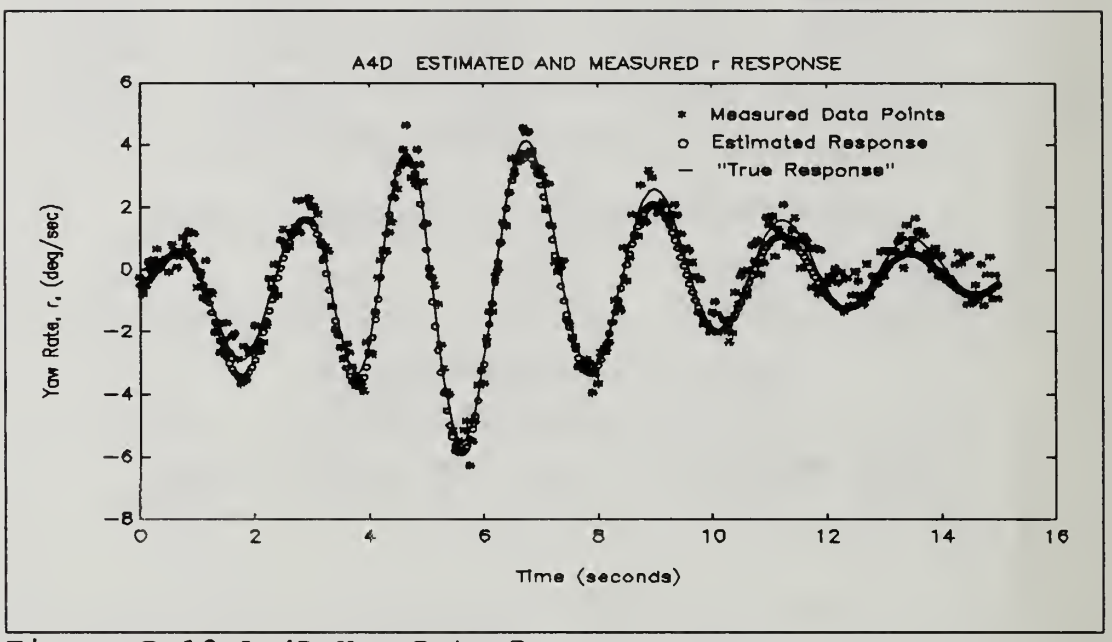

Figure B-18 A-4D Yaw Rate Response

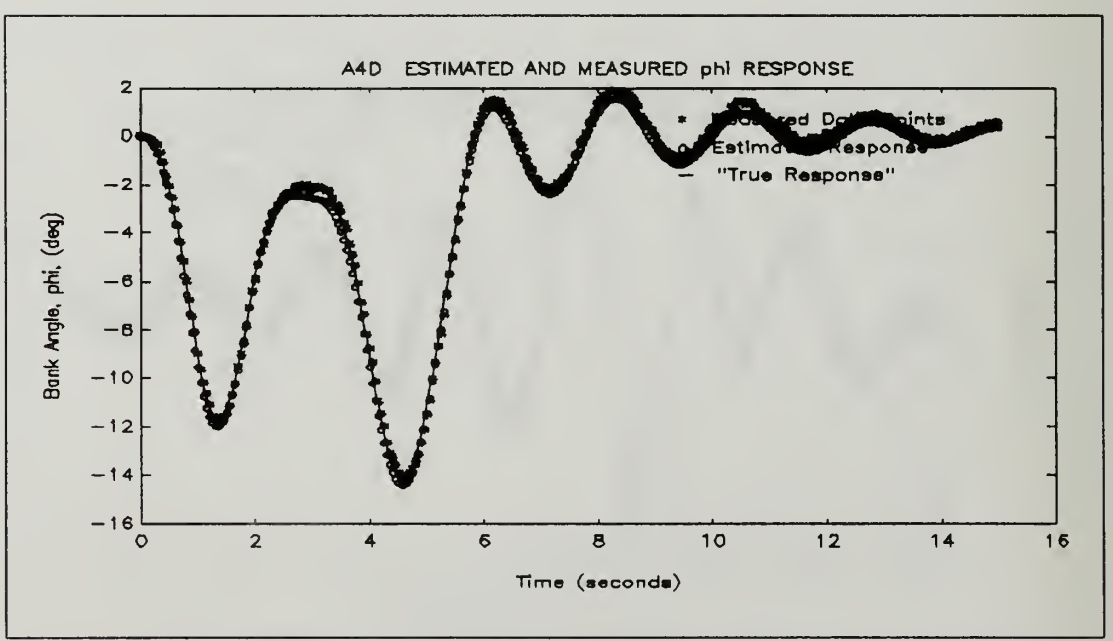

Figure B-19 A-4D Bank Angle Response

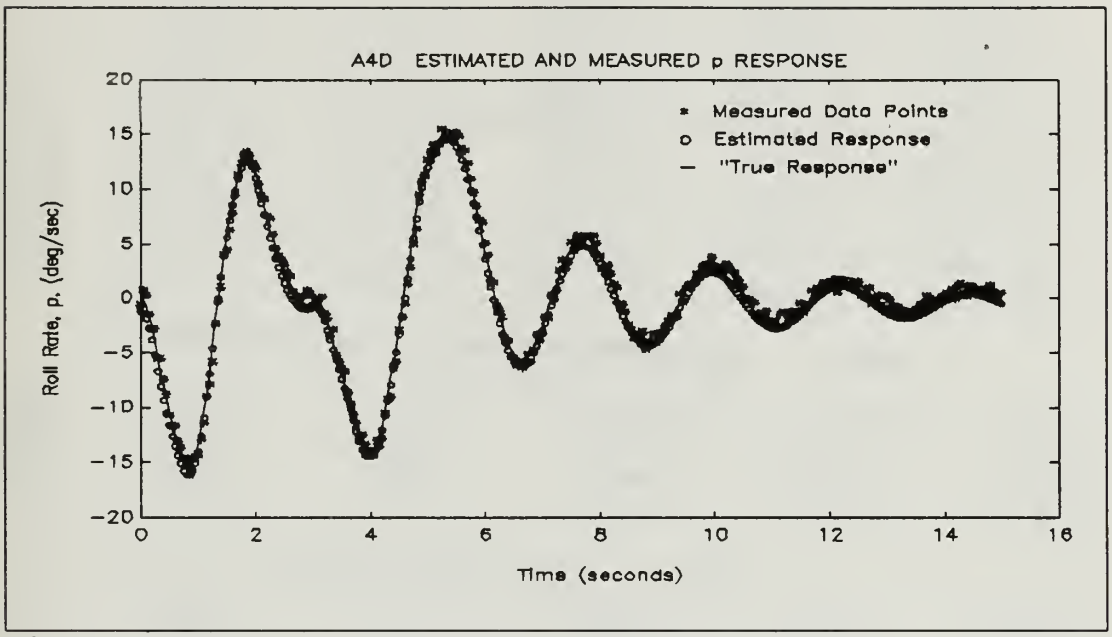

Figure B-20 A-4D Roll Rate Response

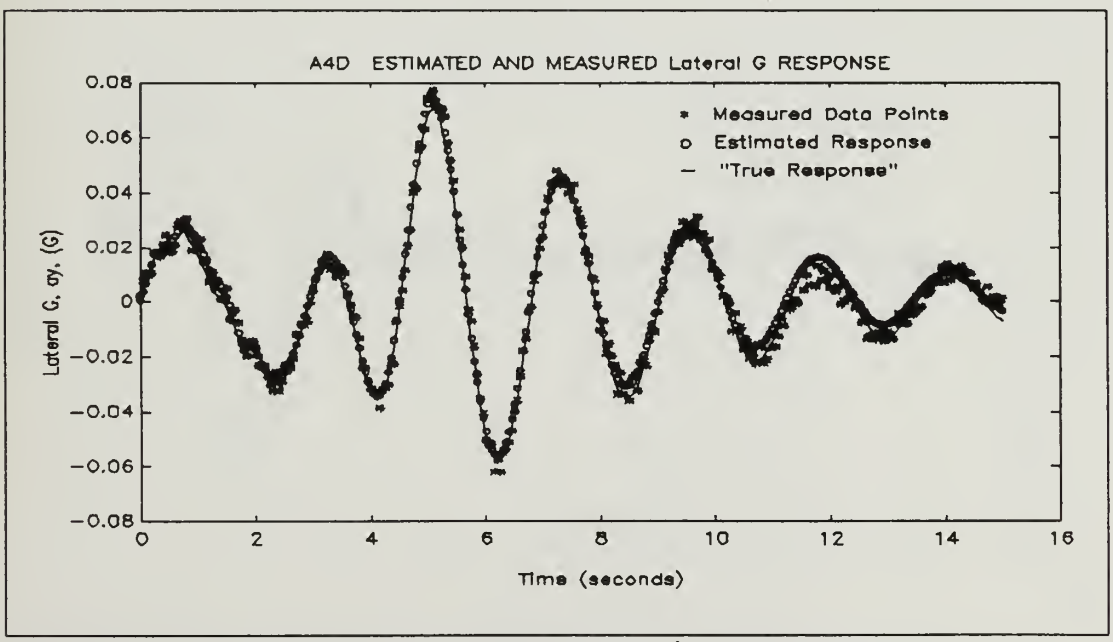

Figure B-21 A-4D Lateral Acceleration Response

A-4D SIMULATED LATERAL-DIRECTIONAL RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	-0.9863	-0.6000	0.0102	0.0273	0.0085
2.0000	-0.1225	-0.1500	0.0011	0.0030	0.0007
3.0000	0.2512	0.2000	0.0011	0.0030	0.0008
4.0000	-0.2610	-0.3500	0.0030	0.0079	0.0013
5.0000	0.0143	-0.0500	0.0044	0.0117	0.0020
6.0000	0.1432	0.1500	0.0085	0.0227	0.0057
7.0000	-0.4134	-0.2000	0.0119	0.0319	0.0074
8.0000	0.0775	0.0500	0.0008	0.0020	0.0004
9.0000	0.0648	-0.0010	0.0012	0.0032	0.0006
10.0000	0.1818	0.1750	0.0072	0.0194	0.0068
11.0000	-0.1035	0.0200	0.0007	0.0019	0.0004
12.0000	0.0340	-0.0750	0.0010	0.0026	0.0006

MMLE STABILITY & CONTROL DERIVATIVES

CY_b	Cl_b	CN_b	Cl_p	CN_p	Cl_r
-0.9863	-0.1225	0.2512	-0.2610	0.0143	0.1432
CN_r	**Clda**	**CNda**	**CYdr**	**Cldr**	**CNdr**
-0.4134	0.0775	0.0648	0.1818	-0.1035	0.0340

INITIAL INPUT DERIVATIVES

-0.6000	-0.1500	0.2000	-0.3500	-0.0500	0.1500
-0.2000	0.0500	-0.0010	0.1750	0.0200	-0.0750

"TRUTH DERIVATIVES" USED TO GENERATE DATA

-0.9800	-0.1200	0.2500	-0.2600	0.0220	0.1400
-0.3500	0.0800	0.0600	0.1700	-0.1050	0.0320

b. NAVION

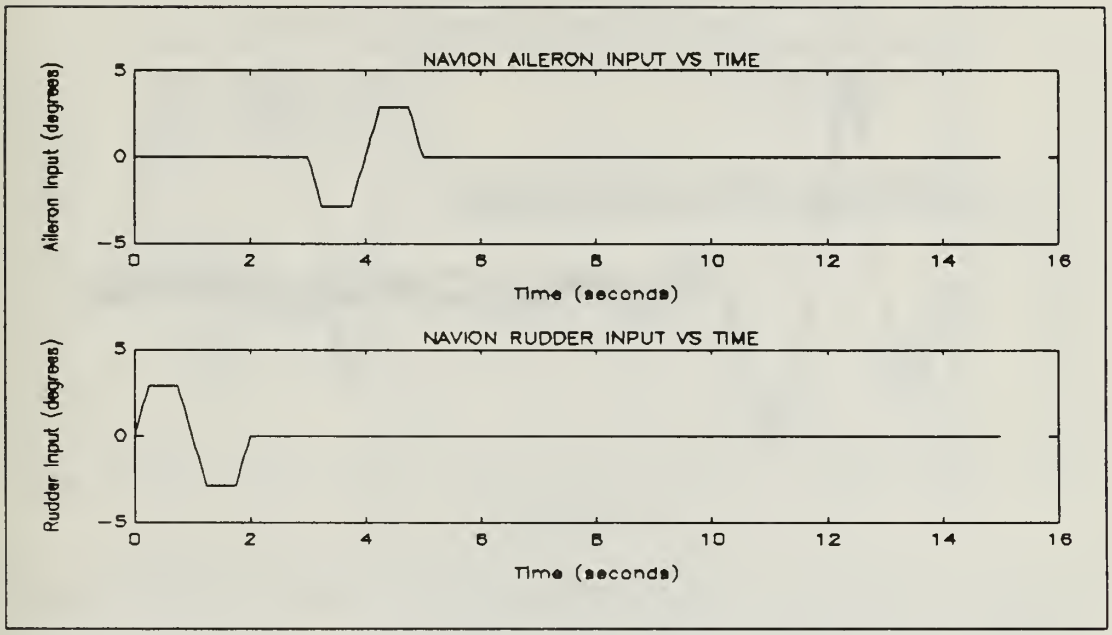

Figure B-22 Navion Aileron and Rudder Inputs

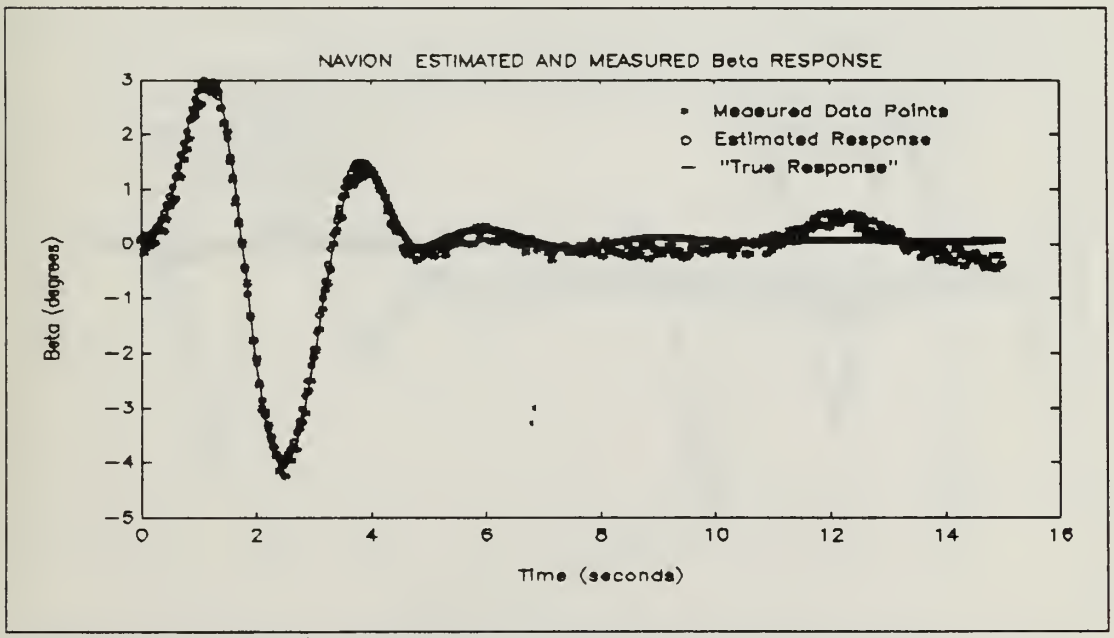

Figure B-23 Navion Beta Response

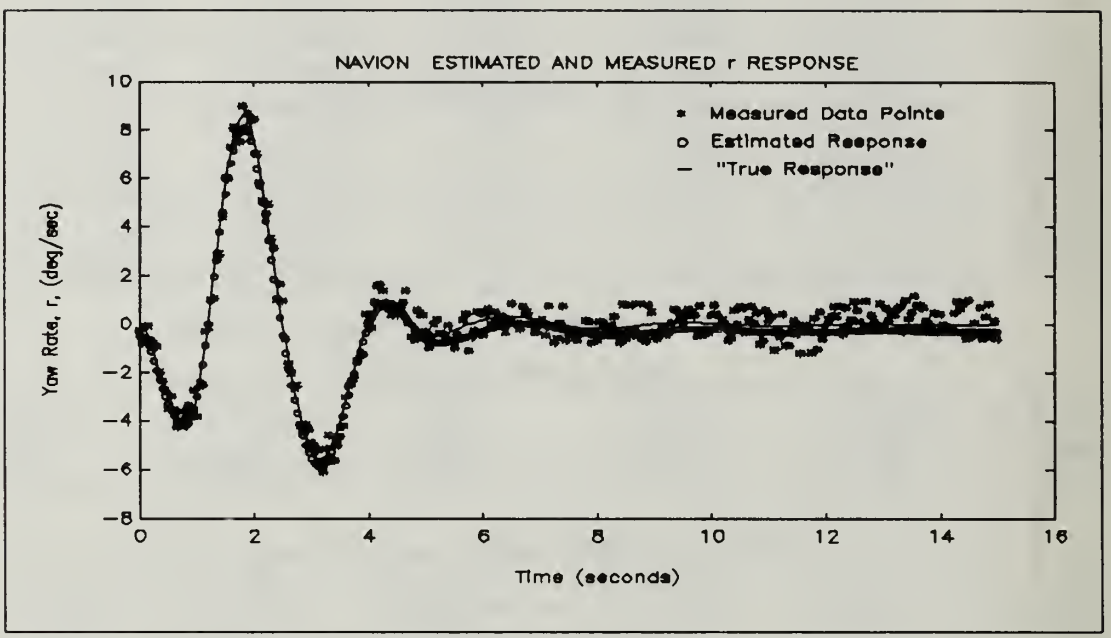

Figure B-24 Navion Yaw Rate Response

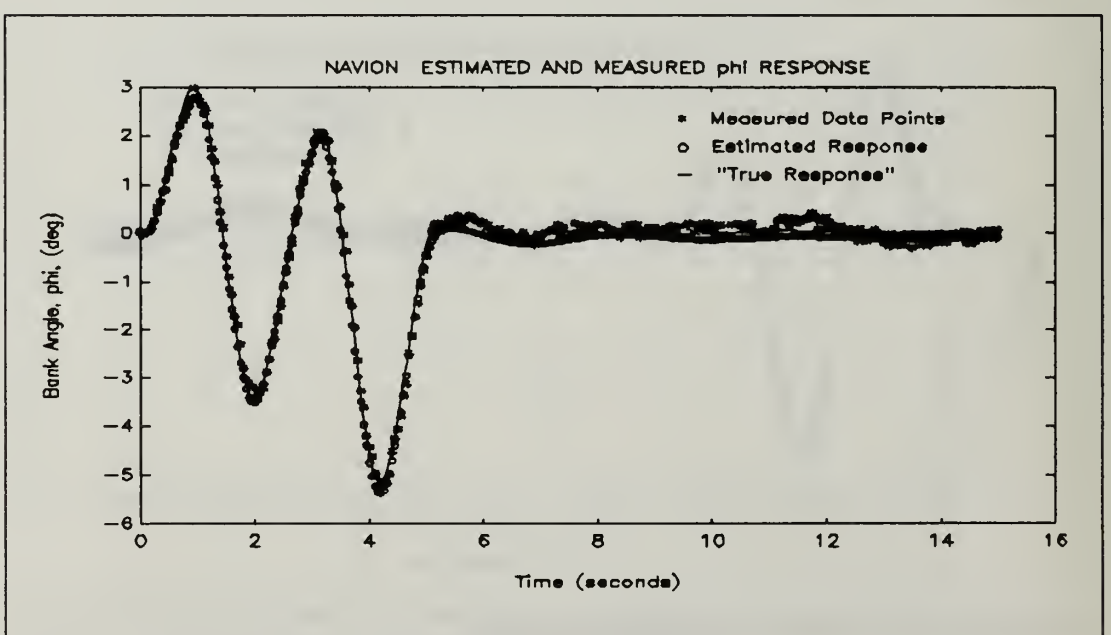

Figure B-25 Navion Bank Angle Response

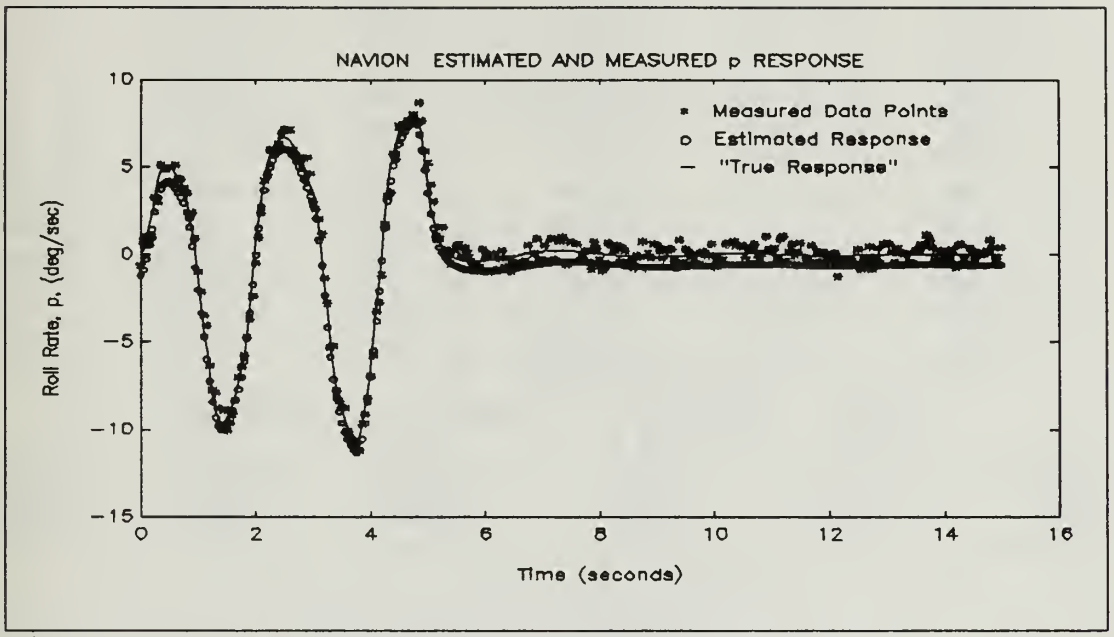

Figure B-26 Navion Roll Rate Response

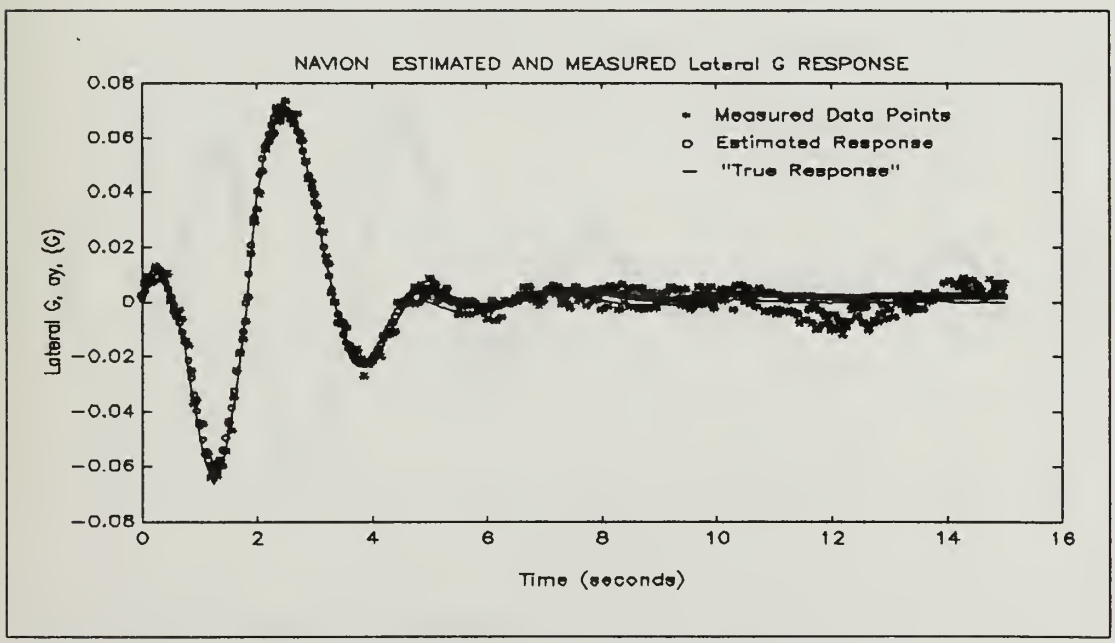

Figure B-27 Navion Lateral Acceleration Response

NAVION SIMULATED LATERAL-DIRECTIONAL RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	-0.5630	-0.6000	0.0059	0.0171	0.0057
2.0000	-0.0746	-0.1500	0.0017	0.0049	0.0006
3.0000	0.0715	0.2000	0.0014	0.0040	0.0004
4.0000	-0.4030	-0.3500	0.0086	0.0249	0.0019
5.0000	-0.0602	-0.0500	0.0070	0.0201	0.0016
6.0000	0.1029	0.1500	0.0055	0.0160	0.0028
7.0000	-0.1189	-0.2000	0.0045	0.0129	0.0019
8.0000	0.1307	0.0500	0.0028	0.0082	0.0009
9.0000	-0.0013	-0.0010	0.0023	0.0068	0.0006
10.0000	0.1611	0.1750	0.0077	0.0222	0.0075
11.0000	0.1031	0.0200	0.0019	0.0055	0.0009
12.0000	-0.0668	-0.0750	0.0019	0.0056	0.0006

MMLE STABILITY & CONTROL DERIVATIVES

CY_b	Cl_b	CN_b	Cl_p	CN_p	Cl_r
-0.5630	-0.0746	0.0715	-0.4030	-0.0602	0.1029
CN_r	**Clda**	**CNda**	**CYdr**	**Cldr**	**CNdr**
-0.1189	0.1307	-0.0013	0.1611	0.1031	-0.0668

INITIAL INPUT DERIVATIVES

-0.6000	-0.1500	0.2000	-0.3500	-0.0500	0.1500
-0.2000	0.0500	-0.0010	0.1750	0.0200	-0.0750

"TRUTH DERIVATIVES" USED TO GENERATE DATA

-0.5640	-0.0740	0.0710	-0.4100	-0.0575	0.1070
-0.1250	0.1340	-0.0035	0.1570	0.1070	-0.0720

c. UAV

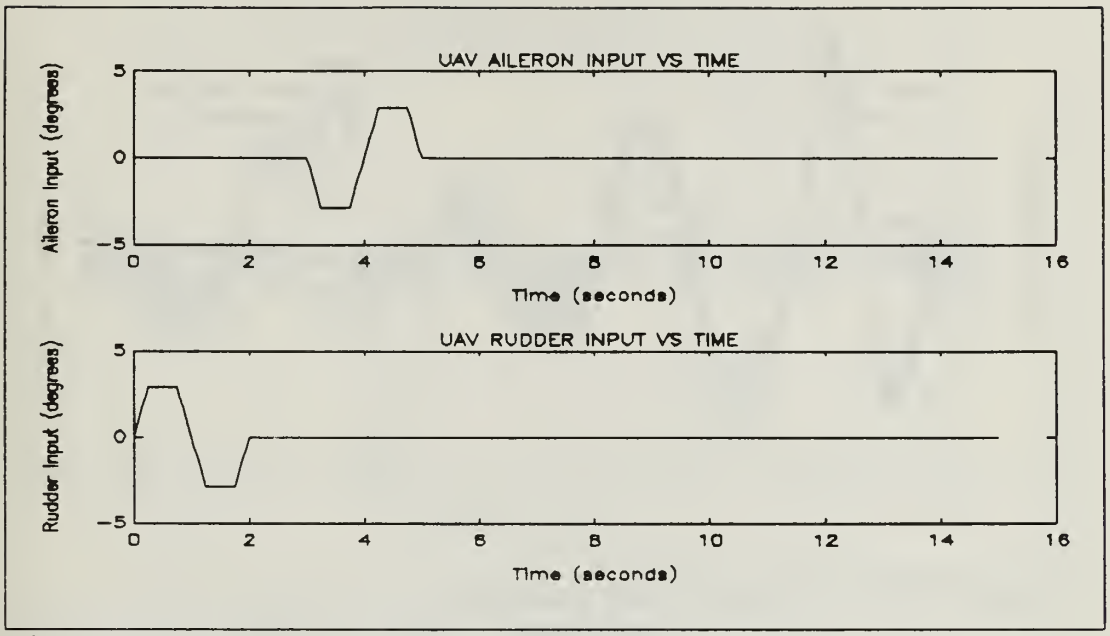

Figure B-28 UAV Aileron and Rudder Inputs

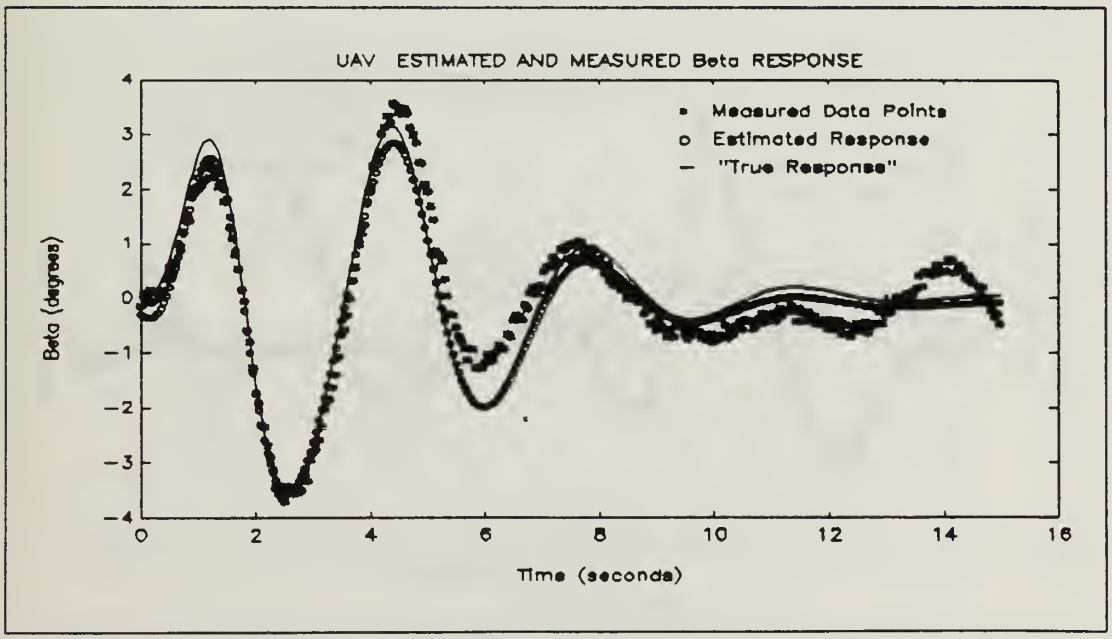

Figure B-29 UAV Beta Response

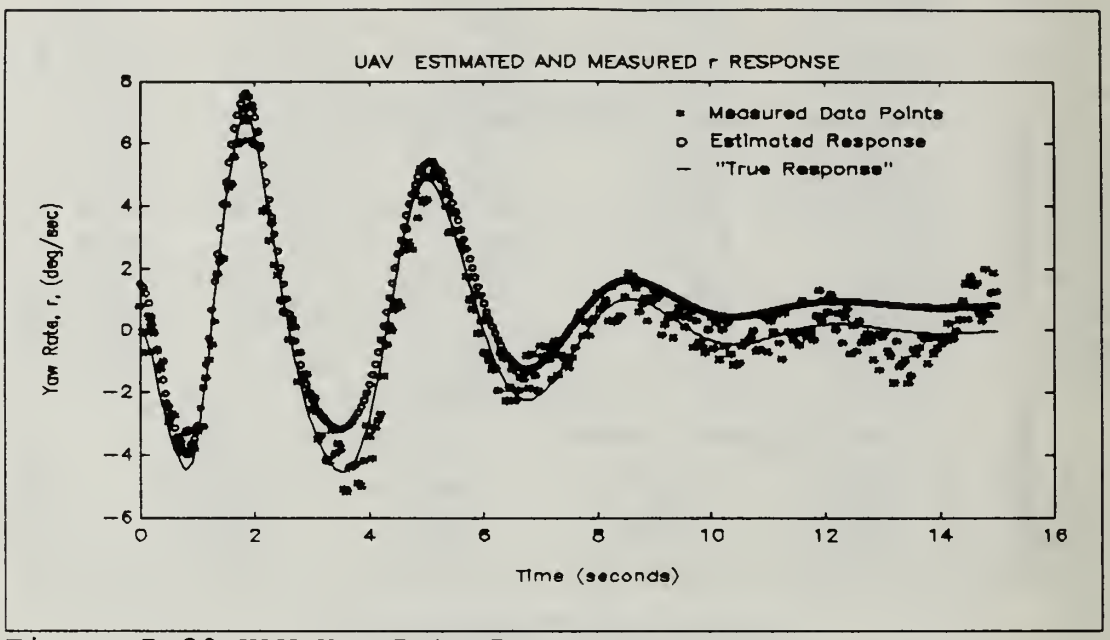

Figure B-30 UAV Yaw Rate Response

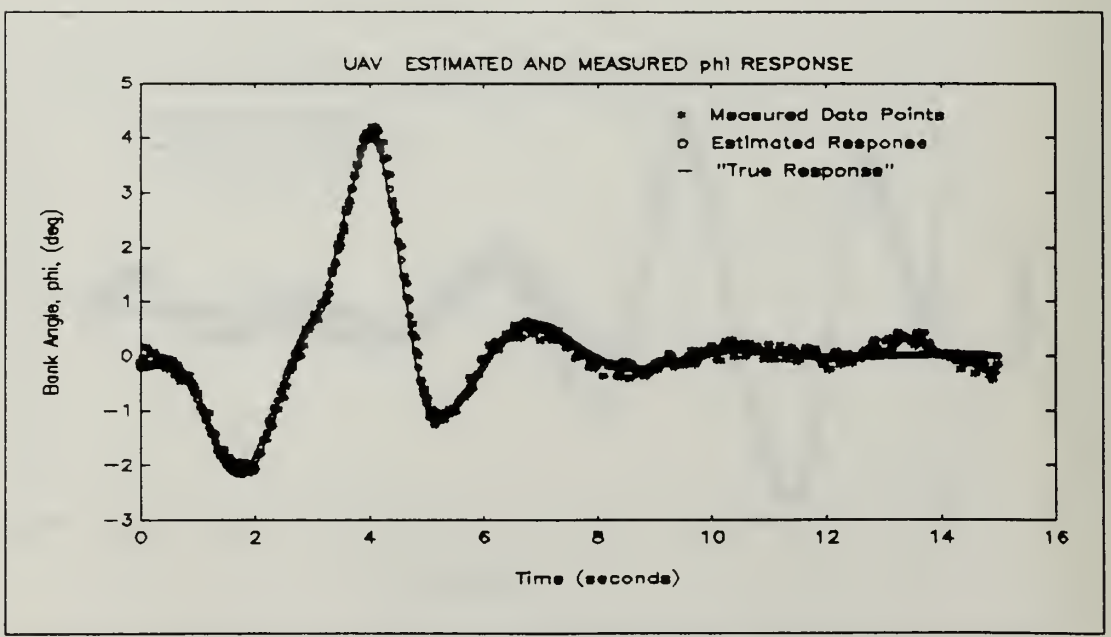

Figure B-31 UAV Bank Angle Response

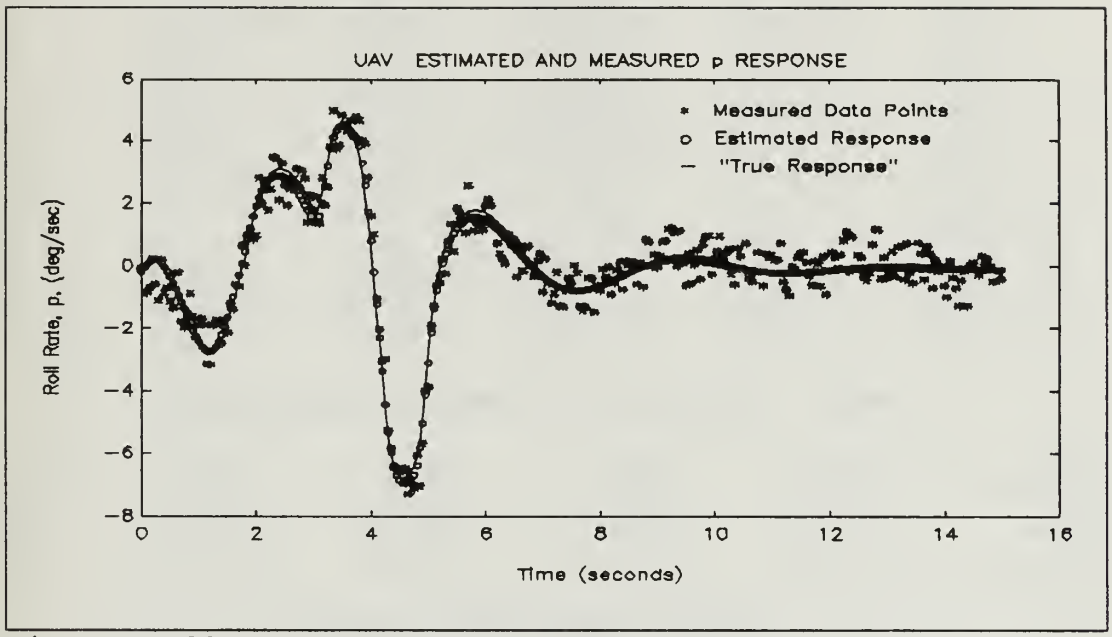

Figure B-32 UAV Roll Rate Response

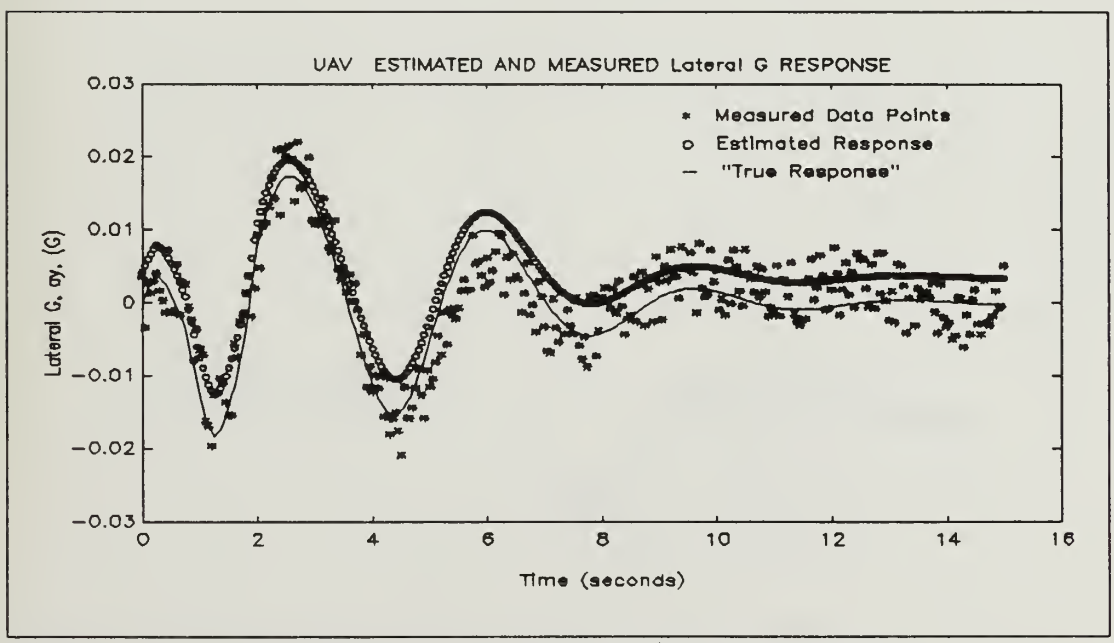

Figure B-33 UAV Lateral Acceleration Response

UAV SIMULATED LATERAL-DIRECTIONAL RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	-0.5140	-0.6000	0.0185	0.0573	0.0164
2.0000	-0.0465	-0.1500	0.0020	0.0062	0.0007
3.0000	0.0551	0.2000	0.0027	0.0082	0.0005
4.0000	-0.5687	-0.3500	0.0211	0.0655	0.0044
5.0000	-0.1551	-0.0500	0.0323	0.1000	0.0038
6.0000	0.1723	0.1500	0.0066	0.0206	0.0036
7.0000	-0.1668	-0.2000	0.0078	0.0241	0.0032
8.0000	-0.1243	0.0500	0.0040	0.0124	0.0012
9.0000	-0.0089	-0.0010	0.0065	0.0202	0.0012
10.0000	0.1334	0.1750	0.0270	0.0838	0.0256
11.0000	0.0086	0.0200	0.0016	0.0051	0.0012
12.0000	-0.0959	-0.0750	0.0020	0.0063	0.0012

MMLE STABILITY & CONTROL DERIVATIVES

CY_b	**Cl_b**	**CN_b**	**Cl_p**	**CN_p**	**Cl_r**
-0.5140	-0.0465	0.0551	-0.5687	-0.1551	0.1723
CN_r	**Clda**	**CNda**	**CYdr**	**Cldr**	**CNdr**
-0.1668	-0.1243	-0.0089	0.1334	0.0086	-0.0959

INITIAL INPUT DERIVATIVES

-0.6000	-0.1500	0.2000	-0.3500	-0.0500	0.1500
-0.2000	0.0500	-0.0010	0.1750	0.0200	-0.0750

"TRUTH DERIVATIVES" USED TO GENERATE DATA

-0.5300	-0.0520	0.0600	-0.6050	-0.0810	0.1650
-0.1620	-0.1250	0.0095	0.1700	0.0120	-0.0920

B. ACTUAL TEST FLIGHT OUTPUT

1. Longitudinal

a. *F-14A*

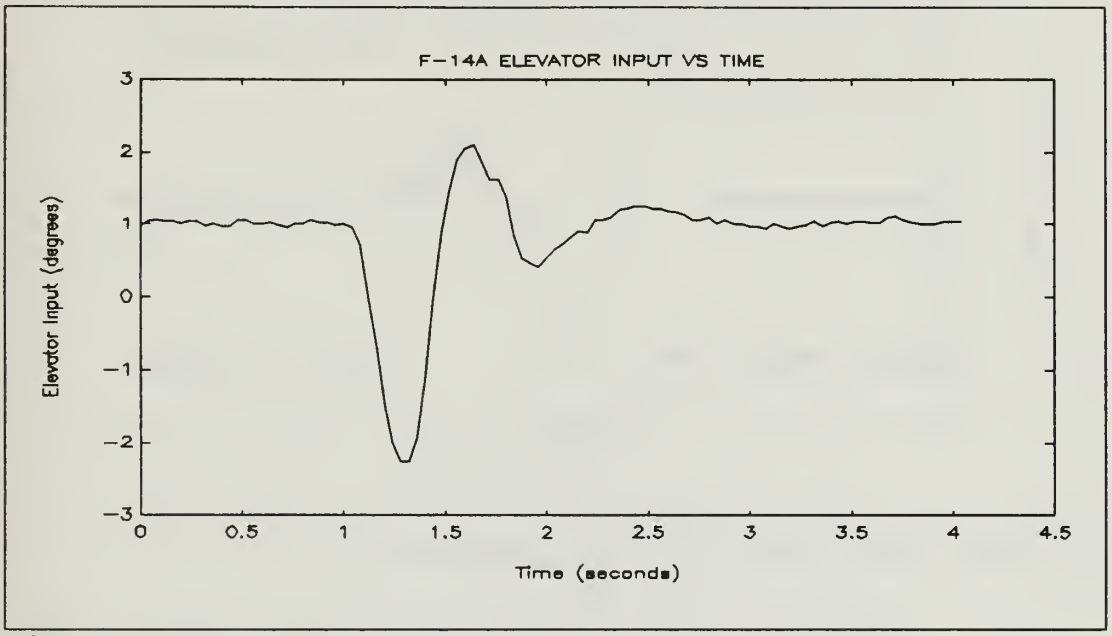

Figure B-34 F-14A Elevator Input

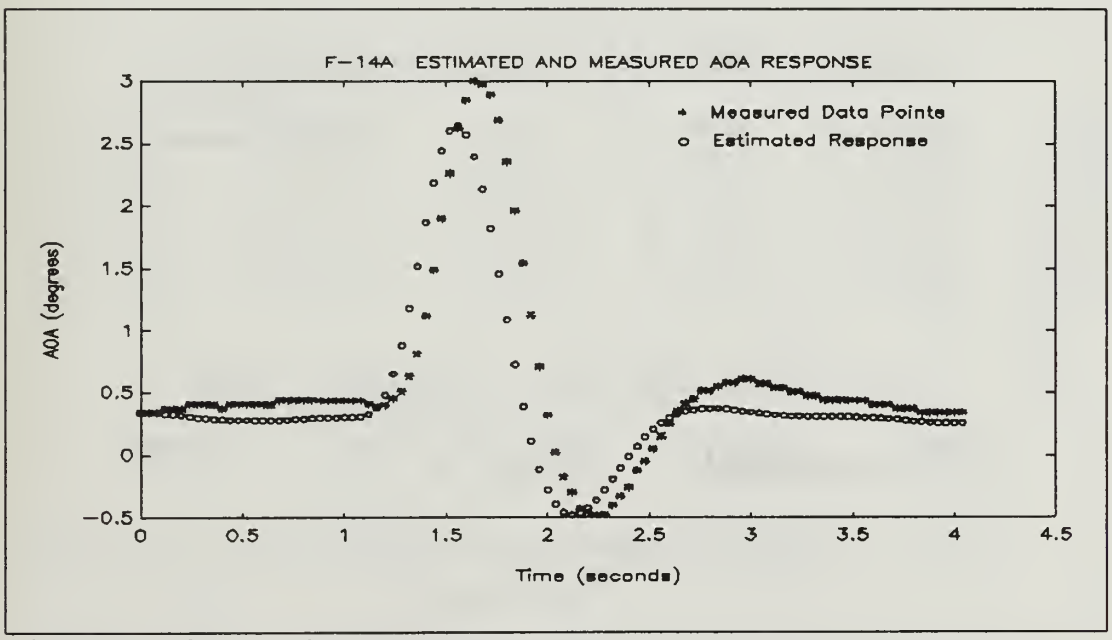

Figure B-35 F-14A AOA Response

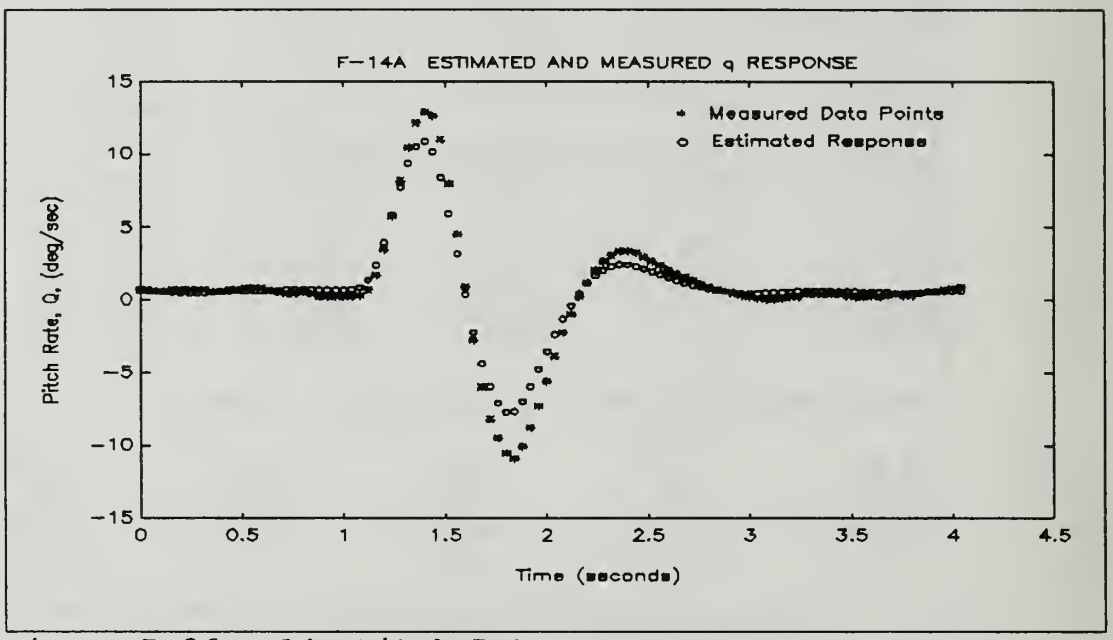

Figure B-36 F-14A Pitch Rate Response

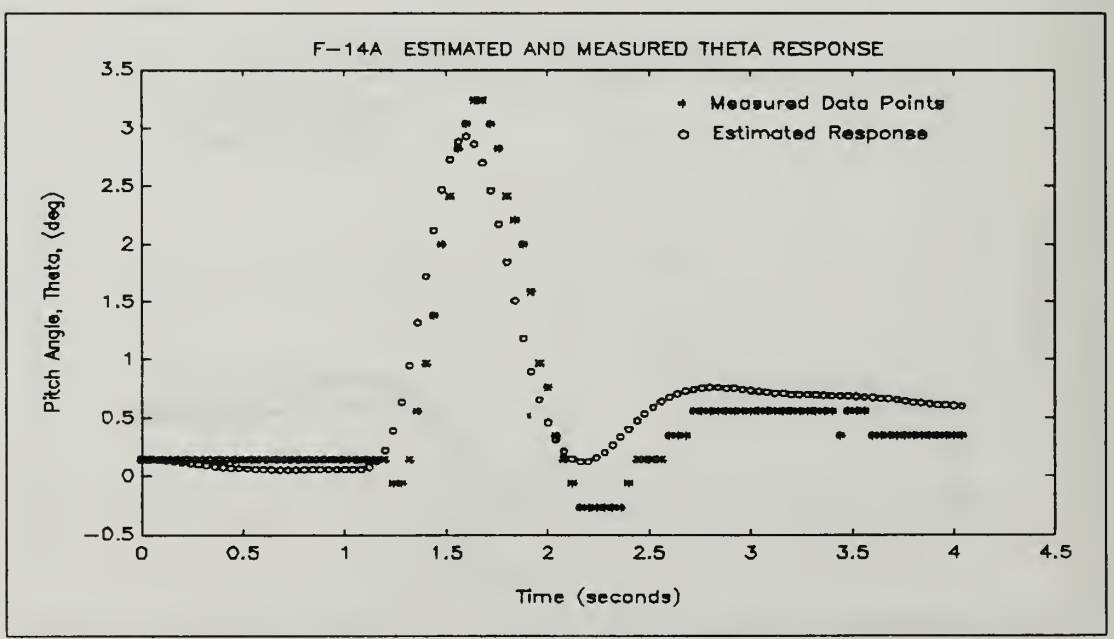

Figure B-37 F-14A Pitch Angle Response

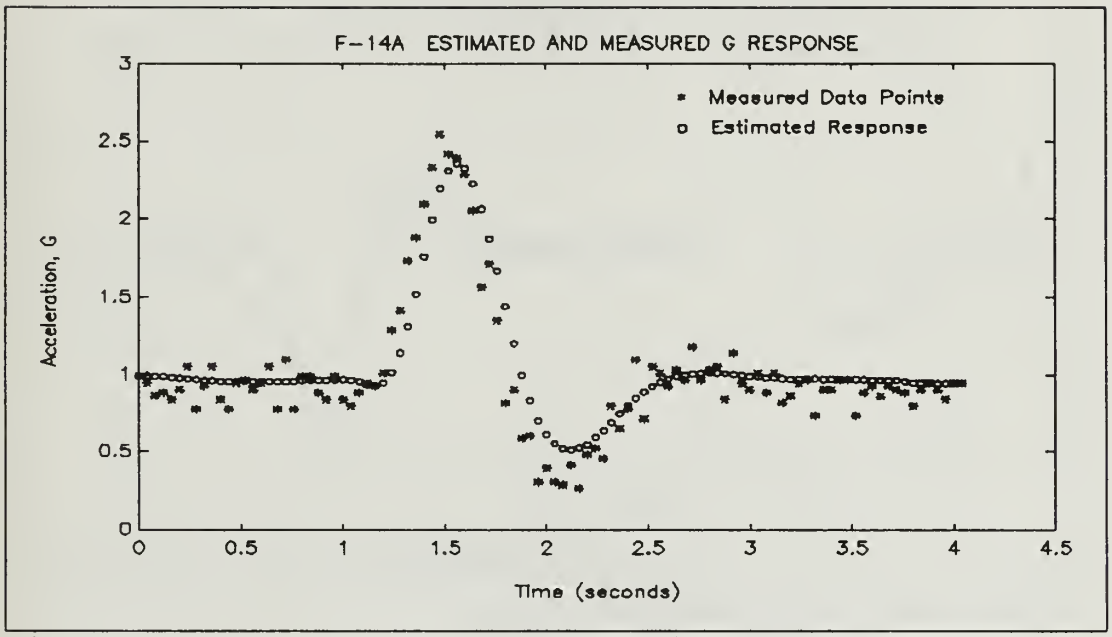

Figure B-38 F-14A Normal Acceleration Response

F-14A LONGITUDINAL FLIGHT TEST RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	6.0109	5.5000	0.1834	0.6366	0.1222
2.0000	-1.5456	-0.6000	0.0119	0.0414	0.0091
3.0000	-24.8107	-18.0000	0.6808	2.3632	0.3380
4.0000	0.5242	0.8000	0.1409	0.4890	0.1125
5.0000	-1.6402	-1.7000	0.0135	0.0469	0.0093

MMLE STABILITY & CONTROL DERIVATIVES

CLA	CMA	CMQ	CLDE	CMDE
6.0109	-1.5456	-24.8107	0.5242	-1.6402

INITIAL INPUT DERIVATIVES

5.5000	-0.6000	-18.0000	0.8000	-1.7000

b. T-37

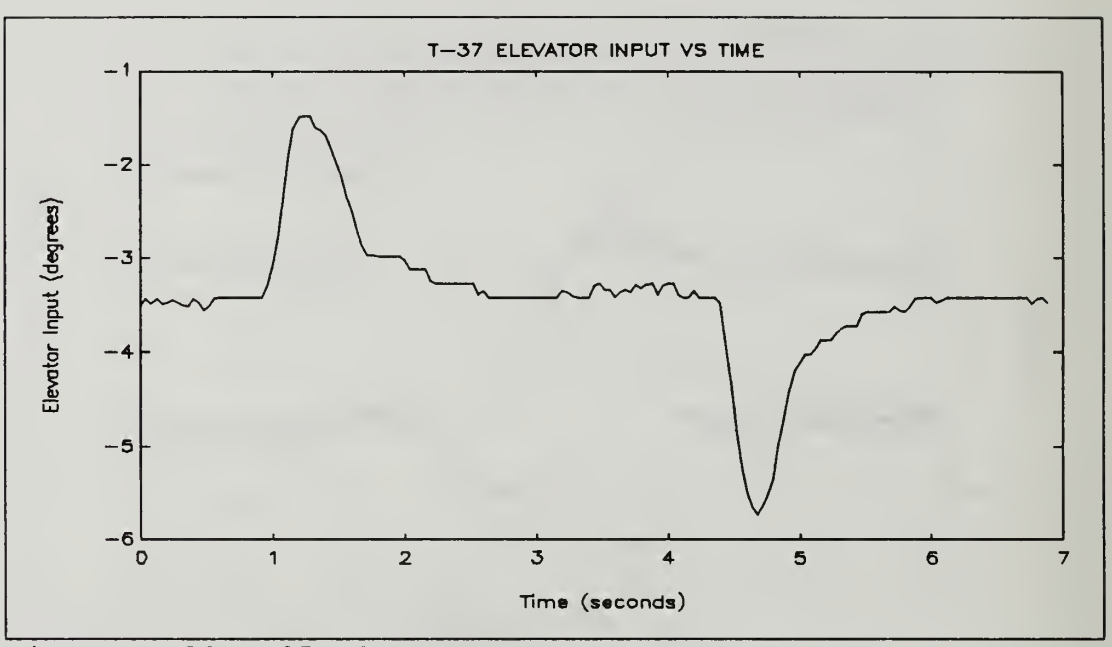

Figure B-39 T-37 Elevator Input

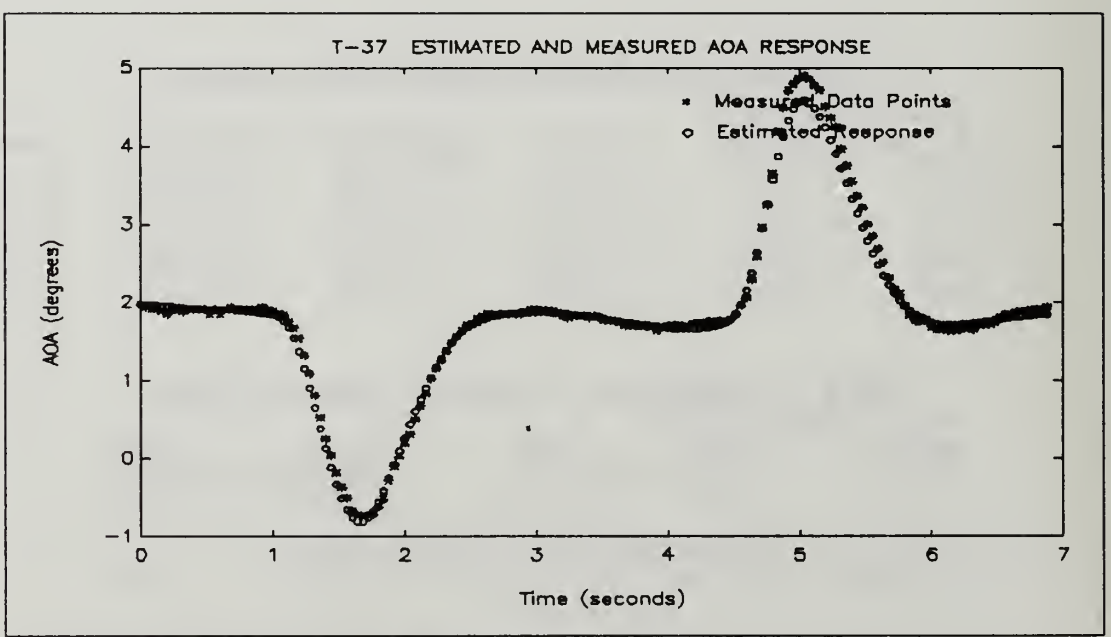

Figure B-40 T-37 AOA Response

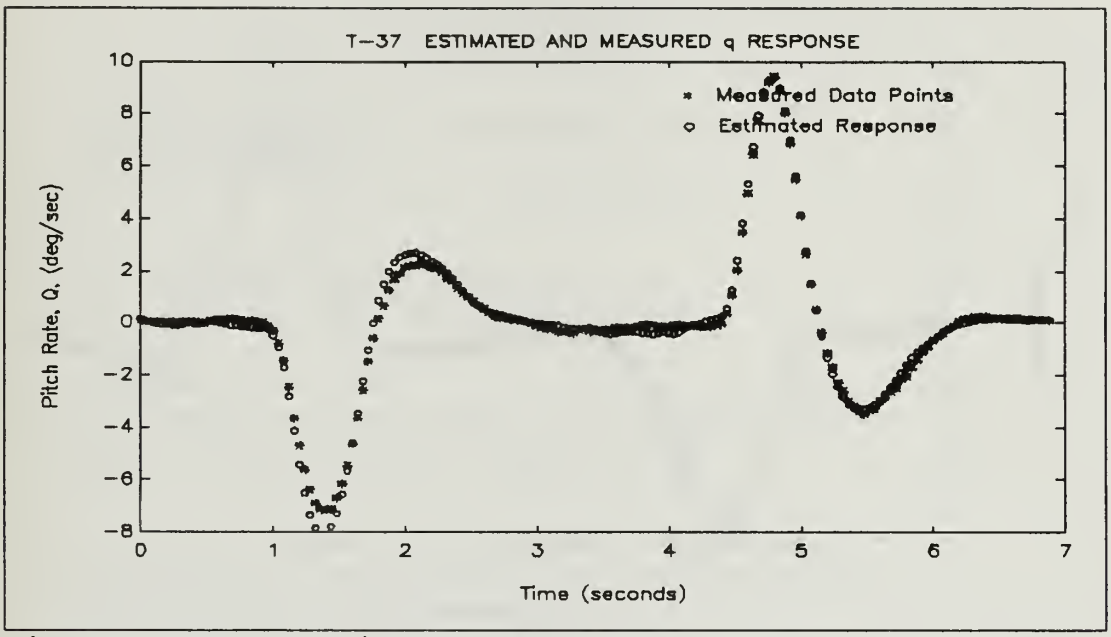

Figure B-41 T-37 Pitch Rate Response

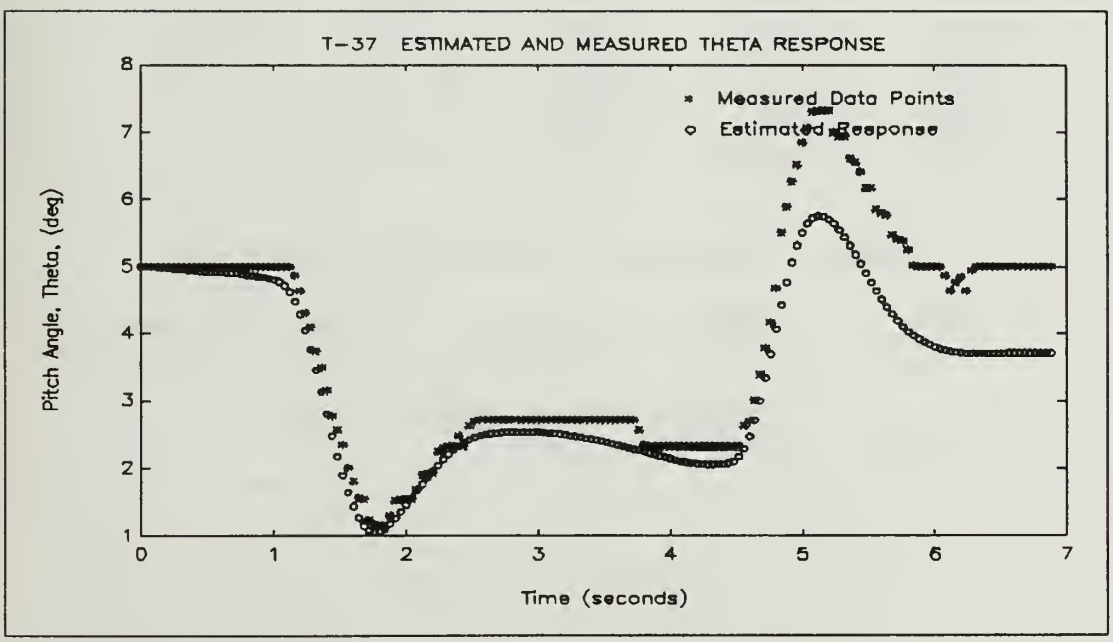

Figure B-42 T-37 Pitch Angle Response

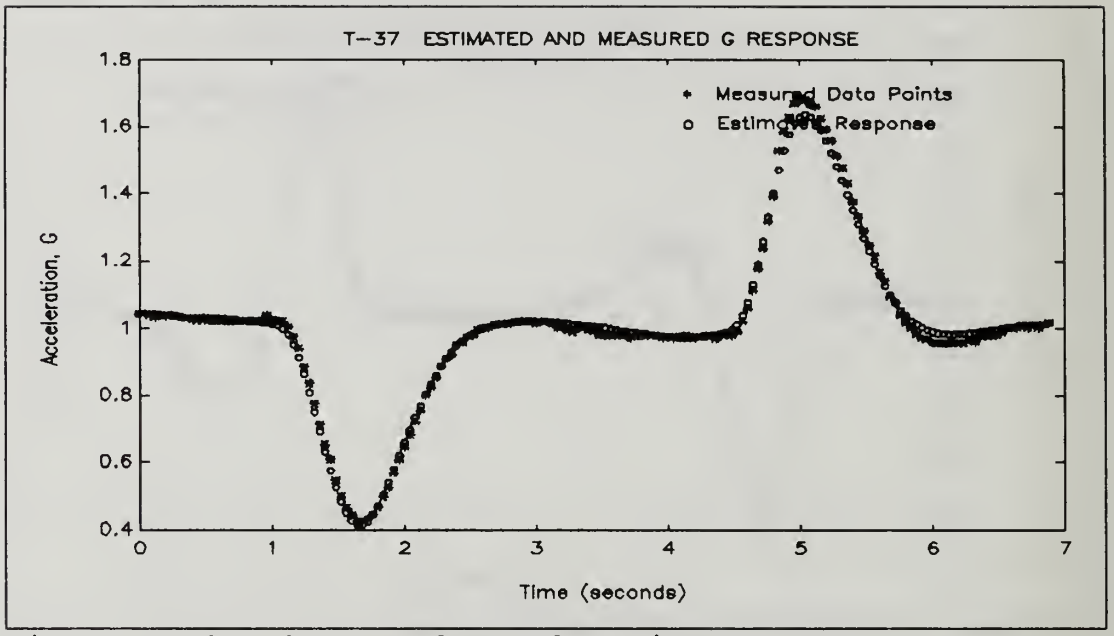

Figure B-43 T-37 Normal Acceleration Response

T-37 LONGITUDINAL FLIGHT TEST RESULTS

pid	p(pid)	pref	cramer	2fcramer	insens
1.0000	7.5725	5.0000	0.0266	0.1052	0.0206
2.0000	-0.9267	-0.5000	0.0071	0.0282	0.0049
3.0000	-32.4972	-20.0000	0.5082	2.0071	0.2574
4.0000	0.1008	0.3500	0.0386	0.1525	0.0311
5.0000	-1.8404	-1.3000	0.0190	0.0749	0.0077

MMLE STABILITY & CONTROL DERIVATIVES

CLA	CMA	CMQ	CLDE	CMDE
7.5725	-0.9267	-32.4972	0.1008	-1.8404

INITIAL INPUT DERIVATIVES

5.0000	-0.5000	-20.0000	0.3500	-1.3000

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5100	2
3.	Chairman, Code AA Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, California 93940-5000	1
4.	Professor Richard M. Howard, Code AA/Ho Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, California 93943-5100	4
5.	Professor Louis V. Schmidt, Code AA/Sc Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, California 93943-5100	1
6.	Lcdr. Robert G. Graham, USN 1169 Paramore Drive Virginia Beach, Virginia 23454	1
7.	Cdr. T.J. Barnes Unmanned Aerial Vehicles Joint Project Code PEO(CU)-UD Washington, DC 20361-1014	1
8.	Mr. Richard J. Foch Ms. Peggy Toot Naval Research Lab, Code 5712 4555 Overlook Ave. SW Washington, DC 20374 .	1
9.	Mr. Paul Heinold Marine Corps UAV Project Office Intelligence Systems (C2IU) Marine Corps Research, Development, and Acquisition Command Quantico, Virginia 22134-5080	1

10. Mr. Jim Murray 1
Room 1021
NASA-Dryden Flight Research Facility
Edwards AFB, California 93523